xebia project:-

Problem Statement:-

HELP International have been able to raise around $ 10 million. Now the CEO of the NGO needs to decide how to use this money strategically and effectively. So, CEO has to make decision to choose the countries that are in the direst need of aid. Hence, your Job as a Data scientist is to categorise the countries using some socio-economic and health factors that determine the overall development of the country. Then you need to suggest the countries which the CEO needs to focus on the most. Find the data here.

Objective:-

To categorise the countries using socio-economic and health factors that determine the overall development of the country.

validating libraries

In [1]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [146…
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from datetime import datetime, timedelta
```

In [147…
```
pd.options.display.float_format='{:.4f}'.format
plt.rcParams['figure.figsize'] = [8,8]
pd.set_option('display.max_columns', 500)
pd.set_option('display.max_colwidth', -1)
sns.set(style='darkgrid')
import matplotlib.ticker as plticker
%matplotlib inline
```

In [148…
```
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.decomposition import IncrementalPCA
from sklearn.neighbors import NearestNeighbors
from random import sample
from numpy.random import uniform
from math import isnan
```

In [149…
```
from sklearn.metrics import silhouette_score
from sklearn.cluster import KMeans
from scipy.cluster.hierarchy import linkage
from scipy.cluster.hierarchy import dendrogram
from scipy.cluster.hierarchy import cut_tree
```

reading the data-

In [150…
```
df1= pd.read_csv('Country-data.csv')
df1.head()
```

Out[150]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	Afghanistan	90.2000	10.0000	7.5800	44.9000	1610	9.4400	56.2000	5.8200	553
1	Albania	16.6000	28.0000	6.5500	48.6000	9930	4.4900	76.3000	1.6500	4090
2	Algeria	27.3000	38.4000	4.1700	31.4000	12900	16.1000	76.5000	2.8900	4460
3	Angola	119.0000	62.3000	2.8500	42.9000	5900	22.4000	60.1000	6.1600	3530
4	Antigua and Barbuda	10.3000	45.5000	6.0300	58.9000	19100	1.4400	76.8000	2.1300	12200

In [151…
```
data_dict = pd.read_csv('data-dictionary.csv')
data_dict.head(10)
```

Out[151]:

	Column Name	Description
0	country	Name of the country
1	child_mort	Death of children under 5 years of age per 1000 live births
2	exports	Exports of goods and services per capita. Given as %age of the GDP per capita
3	health	Total health spending per capita. Given as %age of GDP per capita
4	imports	Imports of goods and services per capita. Given as %age of the GDP per capita
5	Income	Net income per person
6	Inflation	The measurement of the annual growth rate of the Total GDP
7	life_expec	The average number of years a new born child would live if the current mortality patterns are to remain the same
8	total_fer	The number of children that would be born to each woman if the current age-fertility rates remain the same.
9	gdpp	The GDP per capita. Calculated as the Total GDP divided by the total population.

In [152…
```
df1.shape
```

Out[152]:
```
(167, 10)
```

In [153…
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 10 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   country     167 non-null    object 
 1   child_mort  167 non-null    float64
 2   exports     167 non-null    float64
 3   health      167 non-null    float64
 4   imports     167 non-null    float64
 5   income      167 non-null    int64  
 6   inflation   167 non-null    float64
 7   life_expec  167 non-null    float64
 8   total_fer   167 non-null    float64
 9   gdpp        167 non-null    int64  
dtypes: float64(7), int64(2), object(1)
memory usage: 13.2+ KB
```

In [154…

```
df1.describe()
```

Out[154]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
count	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000
mean	38.2701	41.1090	6.8157	46.8902	17144.6886	7.7818	70.5557	2.9480	12964.1557
std	40.3289	27.4120	2.7468	24.2096	19278.0677	10.5707	8.8932	1.5138	18328.7048
min	2.6000	0.1090	1.8100	0.0659	609.0000	-4.2100	32.1000	1.1500	231.0000
25%	8.2500	23.8000	4.9200	30.2000	3355.0000	1.8100	65.3000	1.7950	1330.0000
50%	19.3000	35.0000	6.3200	43.3000	9960.0000	5.3900	73.1000	2.4100	4660.0000
75%	62.1000	51.3500	8.6000	58.7500	22800.0000	10.7500	76.8000	3.8800	14050.0000
max	208.0000	200.0000	17.9000	174.0000	125000.0000	104.0000	82.8000	7.4900	105000.0000

2.cleaning the data

checking missing values

In [155…

```
print('Total missing values in the data')
print('-'*50)
print(df1.isnull().sum())
```

```
Total missing values in the data
--------------------------------------------------
country       0
child_mort    0
exports       0
health        0
imports       0
income        0
inflation     0
life_expec    0
total_fer     0
gdpp          0
dtype: int64
```

converting columns to actual values

In [156…

```
df1['exports'] =df1['exports'] * df1['gdpp']/100
df1['imports'] = df1['imports'] * df1['gdpp']/100
df1['health'] = df1['health'] * df1['gdpp']/100
```

In [157…

```
df1.head()
```

Out[157]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	Afghanistan	90.2000	55.3000	41.9174	248.2970	1610	9.4400	56.2000	5.8200	553
1	Albania	16.6000	1145.2000	267.8950	1987.7400	9930	4.4900	76.3000	1.6500	4090
2	Algeria	27.3000	1712.6400	185.9820	1400.4400	12900	16.1000	76.5000	2.8900	4460
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900	22.4000	60.1000	6.1600	3530
4	Antigua and Barbuda	10.3000	5551.0000	735.6600	7185.8000	19100	1.4400	76.8000	2.1300	12200

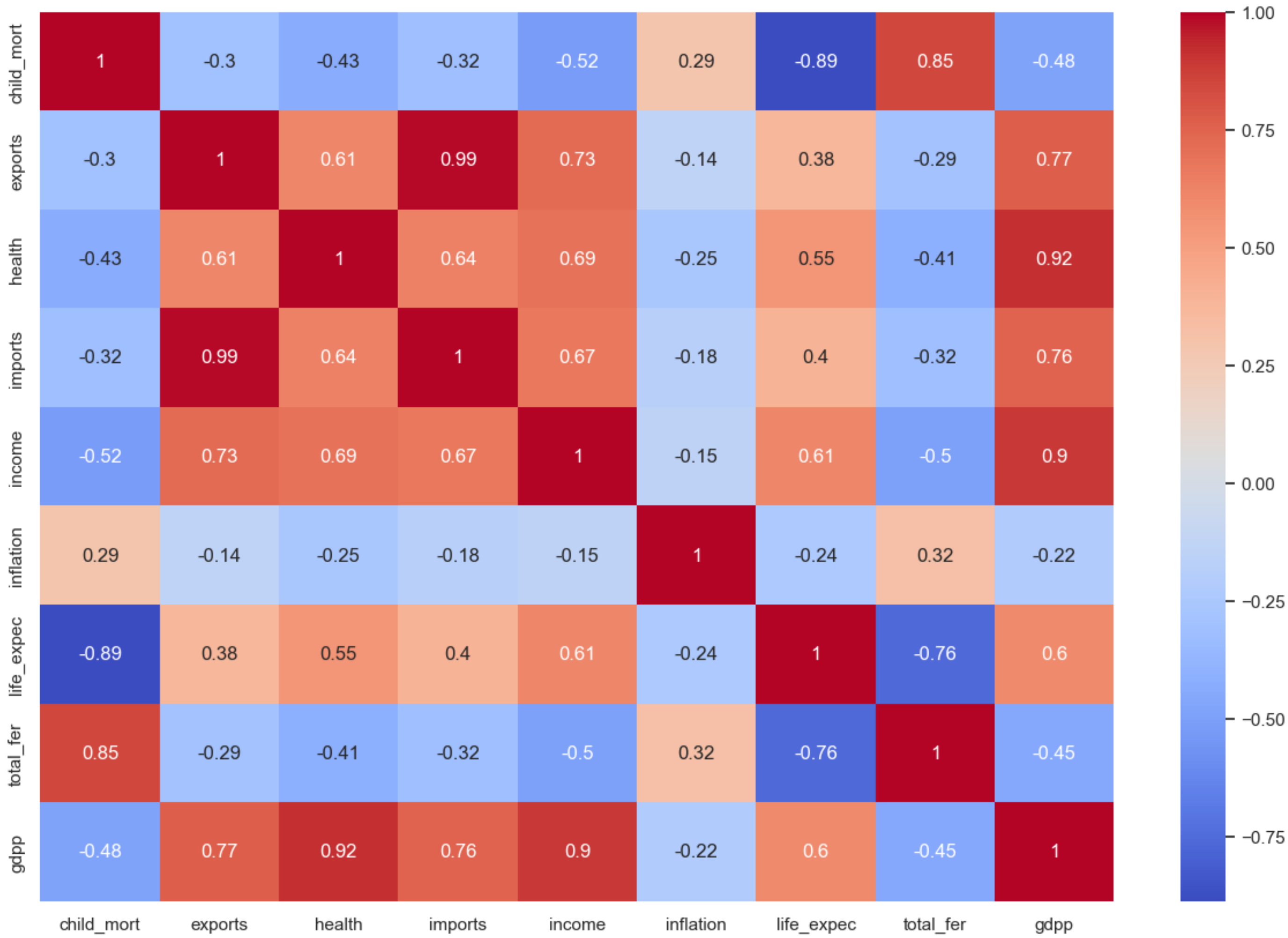
data visualisation

heat map

In [158…

```
plt.figure(figsize = (15,10))
sns.heatmap(df1.corr(),annot = True,cmap='coolwarm')

plt.show()
```



income vs Child Mortality

In [159…

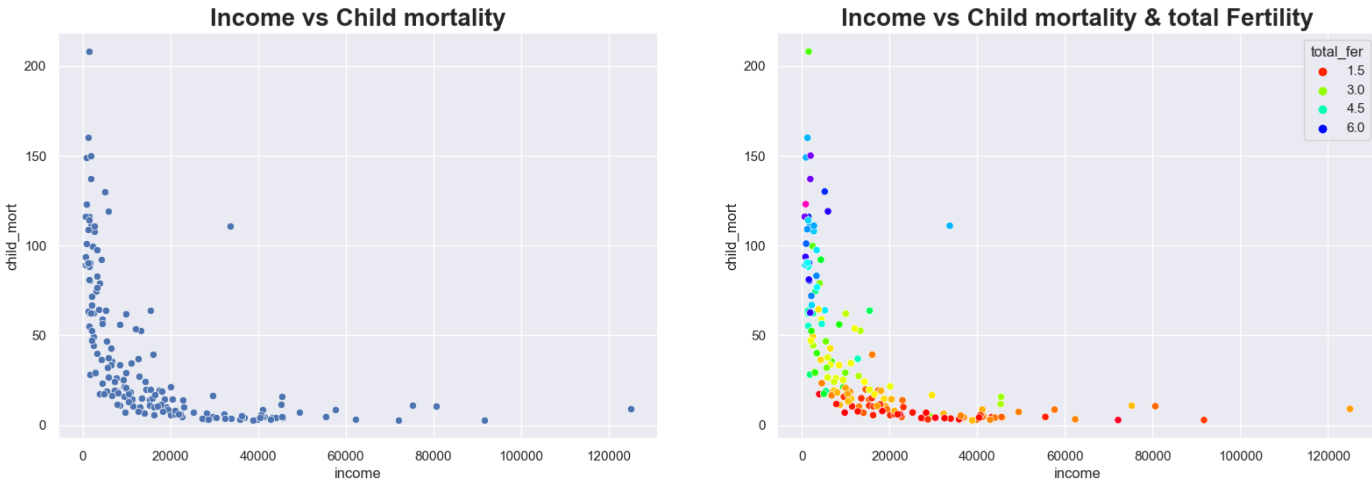
```
plt.figure(figsize=(14, 6))

., 2, 1)
```

Loading [MathJax]/extensions/Safe.js

```
sns.scatterplot(x='income',y='child_mort', data=df1)
plt.title('Income vs Child mortality',fontweight="bold", size=20)

plt.subplot(1, 2, 2)
sns.scatterplot(x='income',y='child_mort',hue='total_fer', data=df1, palette='gist_rainb
plt.title('Income vs Child mortality & total Fertility',fontweight="bold", size=20)
plt.subplots_adjust(right=1.2)
plt.show()
```



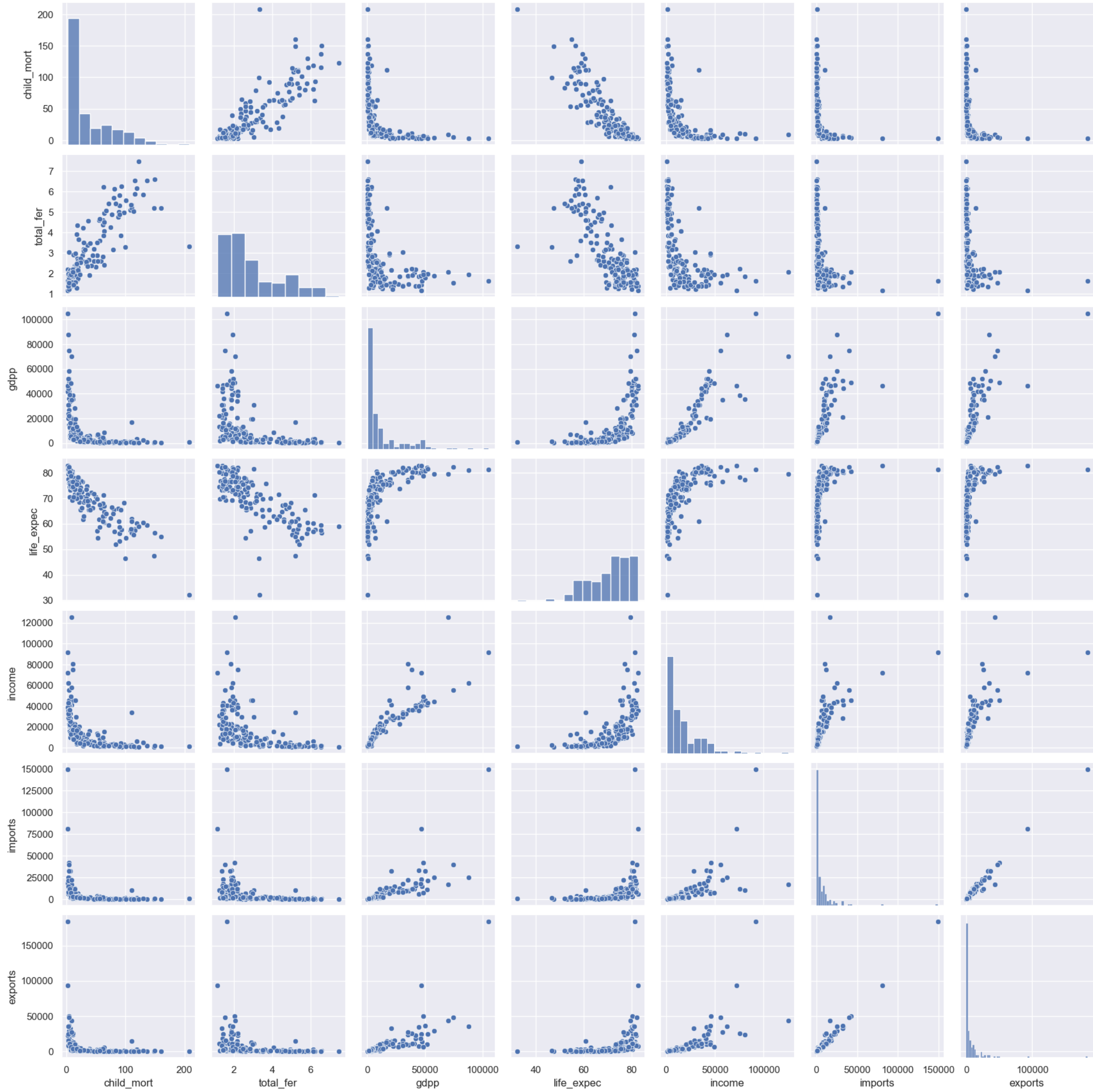
From the plots above We can see that low income people have high child mortality, which means death of children under age 5 is more, where there is a low income Where the income is more we can see there is no mortality In the second plot we can see that, high fertility rate for a woman and low income have high child mortality

pairplot

In [160…

```
sns.pairplot(df1, vars=["child_mort", 'total_fer','gdpp','life_expec','income', 'imports
plt.show()
```

Loading [MathJax]/extensions/Safe.js



country VS child mortality

In [161…

```
Country= df1.groupby('country').child_mort.sum().sort_values(ascending=False)
Country=pd.DataFrame(Country)
Country1=Country.head()
Country2=Country.tail()
display(Country1.head())
print('*'*50)
display(Country2.tail())
```

	child_mort
country	
Haiti	208.0000
Sierra Leone	160.0000
Chad	150.0000
Central African Republic	149.0000
Mali	137.0000

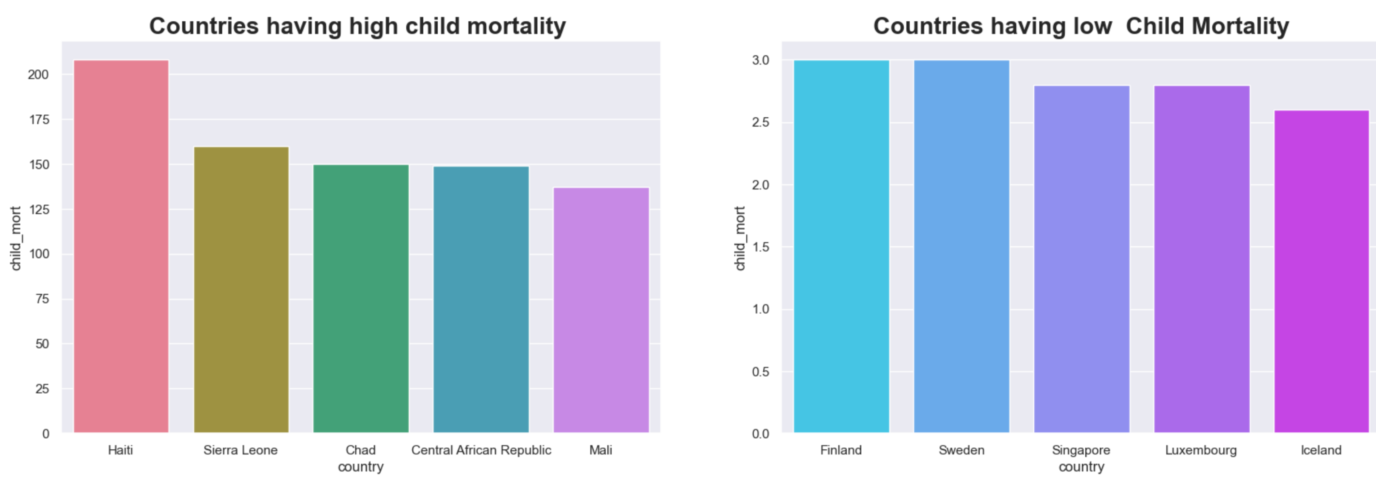
```
**************************************************
```

	child_mort
country	
Finland	3.0000
Sweden	3.0000
Singapore	2.8000
Luxembourg	2.8000
Iceland	2.6000

countries with high & low child mortality

In [162…

```
plt.figure(figsize=(20, 6))
plt.subplot(1,2,1)
sns.barplot(Country1.index, Country1.child_mort, palette='husl')
plt.title('Countries having high child mortality ',fontweight="bold", size=20)
plt.subplot(1,2,2)
sns.barplot(Country2.index, Country2.child_mort, palette='cool')
plt.title('Countries having low  Child Mortality',fontweight="bold", size=20)
plt.show()
```



country VS income

In [163…

```
Income= df1.groupby('country').income.sum().sort_values(ascending=False)
Income=pd.DataFrame(Income)
Income1=Income.head()
Income2=Income.tail()
display(Income1)
```

Loading [MathJax]/extensions/Safe.js

```
print('*'* 50)
display(Income2)
```

	income
country	
Qatar	125000
Luxembourg	91700
Brunei	80600
Kuwait	75200
Singapore	72100

```
**************************************************
```

	income
country	
Central African Republic	888
Niger	814
Burundi	764
Liberia	700
Congo, Dem. Rep.	609

countries with high and low income

plt.figure(figsize=(20, 6)) plt.subplot(1,2,1) sns.barplot(Income1.index, Income1.income, palette='cool') plt.title('Countries with high Income',fontweight="bold", size=20) plt.subplot(1,2,2) sns.barplot(Income2.index, Income2.income, palette='magma') plt.title('Countries with low Income',fontweight="bold", size=20) plt.show()

country VS GDP

In [164…

```
GDP= df1.groupby('country').gdpp.sum().sort_values(ascending=False)
GDP=pd.DataFrame(GDP)
GDP1=GDP.head()
GDP2=GDP.tail()
display(GDP1)
print('*'* 50)
display(GDP2)
```

	gdpp
country	
Luxembourg	105000
Norway	87800
Switzerland	74600
Qatar	70300
Denmark	58000

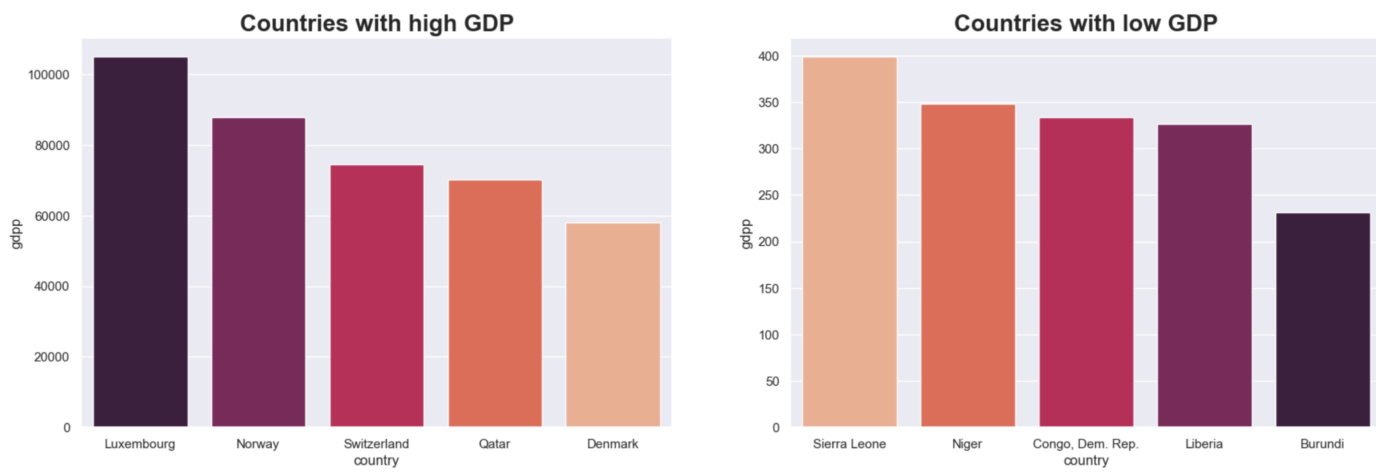
```
**************************************************
```

country	gdpp
Sierra Leone	399
Niger	348
Congo, Dem. Rep.	334
Liberia	327
Burundi	231

countries with high and low GDP

In [165…

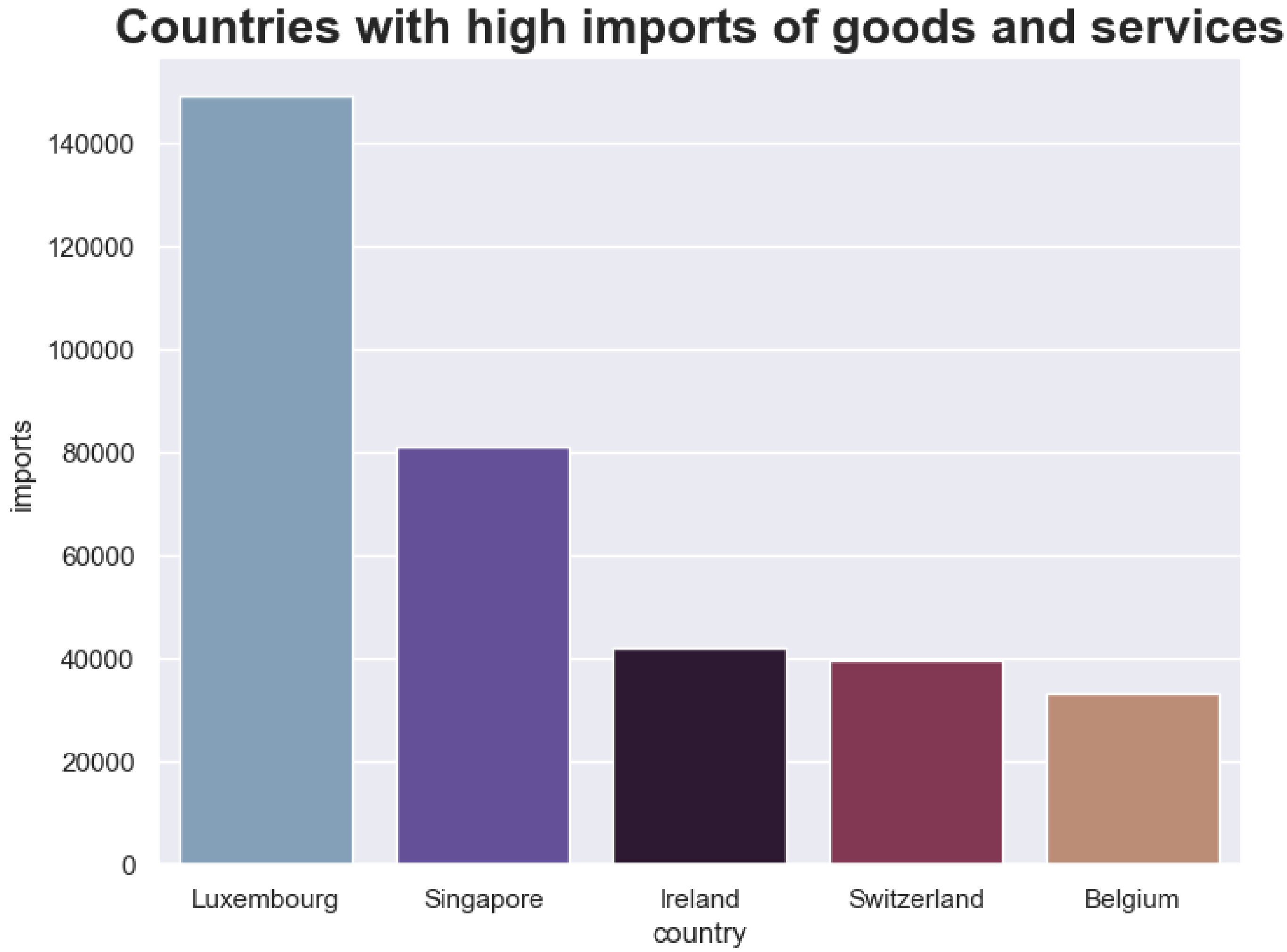
```
plt.figure(figsize=(20, 6))
plt.subplot(1,2,1)
sns.barplot(GDP1.index, GDP1.gdpp, palette='rocket')
plt.title('Countries with high GDP',fontweight="bold", size=20)
plt.subplot(1,2,2)
sns.barplot(GDP2.index, GDP2.gdpp, palette='rocket_r')
plt.title('Countries with low GDP',fontweight="bold", size=20)
plt.show()
```



counties VS import

In [166…

```
plt.figure(figsize=(8, 6))
Imports=df1.groupby('country').imports.sum().sort_values(ascending=False)
Imports= pd.DataFrame(Imports)
Imports1=Imports.head()
sns.barplot(Imports1.index,Imports1.imports, palette='twilight')
plt.title('Countries with high imports of goods and services',fontweight="bold", size=20
plt.show()
display(Imports1)
Imports2=Imports.tail(2)
display(Imports2)
```



country	imports
Luxembourg	149100.0000
Singapore	81084.0000
Ireland	42125.5000
Switzerland	39761.8000
Belgium	33166.8000

country	imports
Burundi	90.5520
Myanmar	0.6511

life EXPECTANCY of countries

In [167…

```
Life_Ex= df1.sort_values(by=['life_expec'],ascending=True)
Life_Ex.head(100)
```

Loading [MathJax]/extensions/Safe.js

Out[167]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
66	Haiti	208.0000	101.2860	45.7442	428.3140	1500	5.4500	32.1000	3.3300	662
87	Lesotho	99.7000	460.9800	129.8700	1181.7000	2380	4.1500	46.5000	3.3000	1170
31	Central African Republic	149.0000	52.6280	17.7508	118.1900	888	2.0100	47.5000	5.2100	446
166	Zambia	83.1000	540.2000	85.9940	451.1400	3280	14.0000	52.0000	5.4000	1460
94	Malawi	90.5000	104.6520	30.2481	160.1910	1030	12.1000	53.1000	5.3100	459
...	...	...	...	...	...	...	...	...	...	...
22	Brazil	19.8000	1198.4000	1009.1200	1321.6000	14500	8.4100	74.2000	1.8000	11200
140	Sri Lanka	11.2000	550.7600	82.6140	753.0800	8560	22.8000	74.4000	2.2000	2810
67	Hungary	6.0000	10715.8000	960.2300	10021.5000	22300	2.3300	74.5000	1.2500	13100
71	Iran	19.3000	1593.3200	365.6800	1266.8200	17400	15.9000	74.5000	1.7600	6530
95	Malaysia	7.9000	7881.8300	398.1730	6439.7000	21100	7.2700	74.5000	2.1500	9070

100 rows × 10 columns

EXPORTS of different countries

In [168…

```python
Exports=df1.sort_values(by=['exports'], ascending= False)
display(Exports[0:8])
```

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdp
91	Luxembourg	2.8000	183750.0000	8158.5000	149100.0000	91700	3.6200	81.3000	1.6300	1050
133	Singapore	2.8000	93200.0000	1845.3600	81084.0000	72100	-0.0460	82.7000	1.1500	466
73	Ireland	4.2000	50161.0000	4475.5300	42125.5000	45700	-3.2200	80.4000	2.0500	487
145	Switzerland	4.5000	47744.0000	8579.0000	39761.8000	55500	0.3170	82.2000	1.5200	746
123	Qatar	9.0000	43796.9000	1272.4300	16731.4000	125000	6.9800	79.5000	2.0700	703
110	Netherlands	4.5000	36216.0000	5985.7000	31990.8000	45500	0.8480	80.7000	1.7900	503
114	Norway	3.2000	34856.6000	8323.4400	25023.0000	62300	5.9500	81.0000	1.9500	878
15	Belgium	4.5000	33921.6000	4750.8000	33166.8000	41100	1.8800	80.0000	1.8600	444

In [169…

```python
Export=df1.sort_values(by=['exports'], ascending= True)
display(Export[0:8])
```

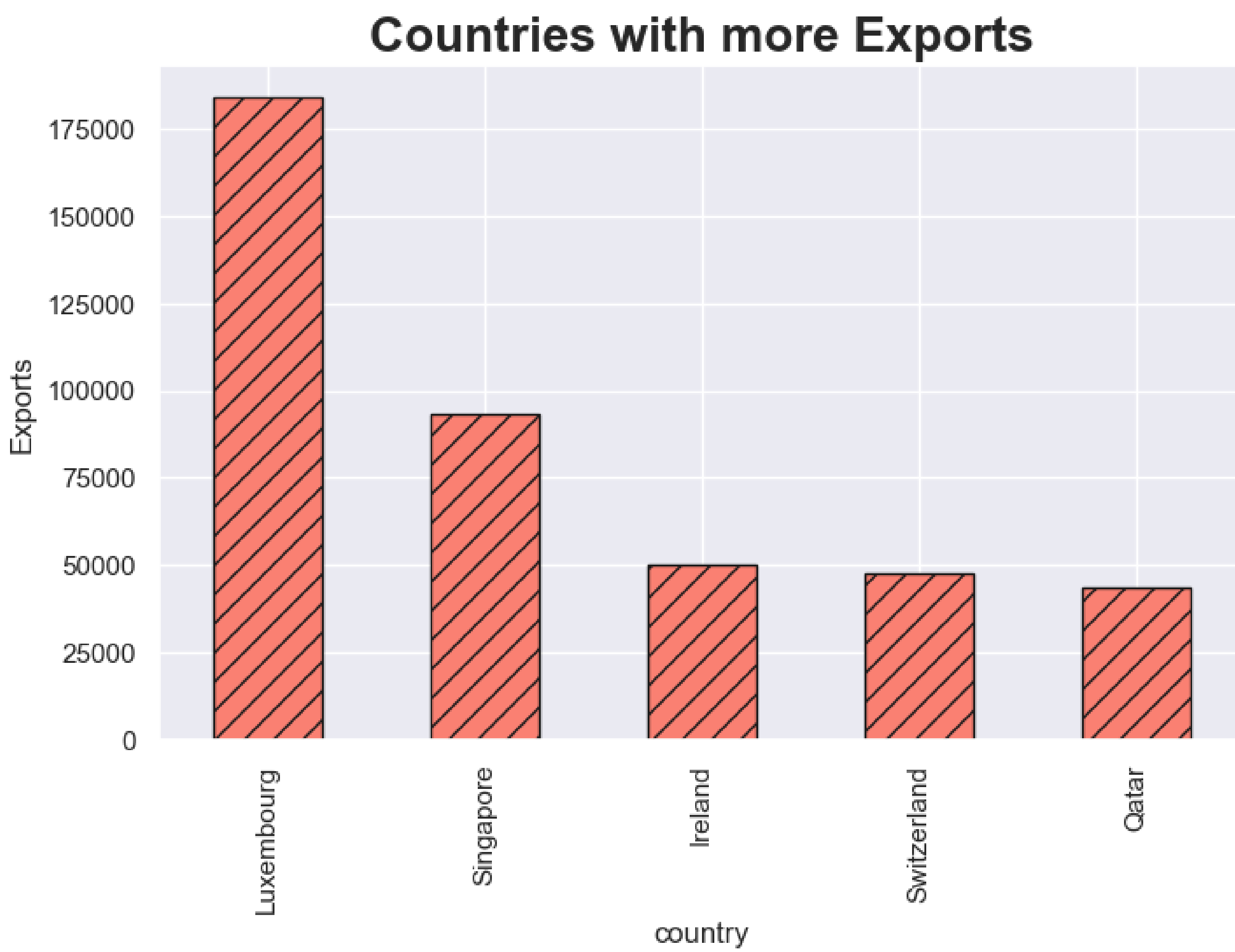
Loading [MathJax]/extensions/Safe.js

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
107	Myanmar	64.4000	1.0769	19.4636	0.6511	3720	7.0400	66.8000	2.4100	988
26	Burundi	93.6000	20.6052	26.7960	90.5520	764	12.3000	57.7000	6.2600	231
50	Eritrea	55.2000	23.0878	12.8212	112.3060	1420	11.6000	61.7000	4.6100	482
31	Central African Republic	149.0000	52.6280	17.7508	118.1900	888	2.0100	47.5000	5.2100	446
0	Afghanistan	90.2000	55.3000	41.9174	248.2970	1610	9.4400	56.2000	5.8200	553
109	Nepal	47.0000	56.7136	31.0800	215.4880	1990	15.1000	68.3000	2.6100	592
88	Liberia	89.3000	62.4570	38.5860	302.8020	700	5.4700	60.8000	5.0200	327
132	Sierra Leone	160.0000	67.0320	52.2690	137.6550	1220	17.2000	55.0000	5.2000	399

countries with high EXPORT

In [170…

```
plt.figure(figsize=(8, 5))
df1.groupby('country').exports.sum().sort_values(ascending=False).head().plot.bar(color=
plt.ylabel('Exports')
plt.title('Countries with more Exports',fontweight="bold", size=20)
plt.show()
```



countries and its HEALTH condition

Loading [MathJax]/extensions/Safe.js

In [171…

```
Health=df1.sort_values(by=['health'], ascending= True)
Health[0:8]
```

Out[171]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
50	Eritrea	55.2000	23.0878	12.8212	112.3060	1420	11.6000	61.7000	4.6100	482
93	Madagascar	62.2000	103.2500	15.5701	177.5900	1390	8.7900	60.8000	4.6000	413
31	Central African Republic	149.0000	52.6280	17.7508	118.1900	888	2.0100	47.5000	5.2100	446
112	Niger	123.0000	77.2560	17.9568	170.8680	814	2.5500	58.8000	7.4900	348
107	Myanmar	64.4000	1.0769	19.4636	0.6511	3720	7.0400	66.8000	2.4100	988
106	Mozambique	101.0000	131.9850	21.8299	193.5780	918	7.6400	54.5000	5.5600	419
116	Pakistan	92.1000	140.4000	22.8800	201.7600	4280	10.9000	65.3000	3.8500	1040
37	Congo, Dem. Rep.	116.0000	137.2740	26.4194	165.6640	609	20.8000	57.5000	6.5400	334

In [172…

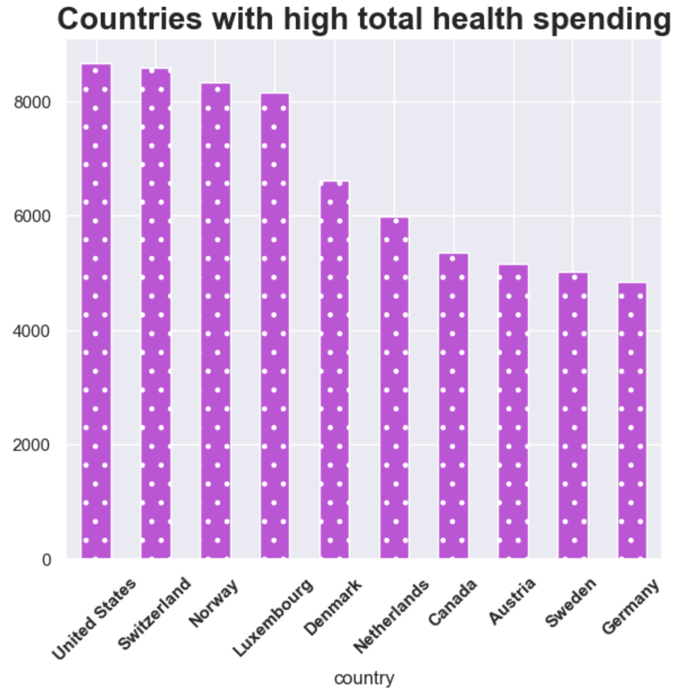
```
Health1=df1.sort_values(by=['health'], ascending= False)
Health1[0:8]
```

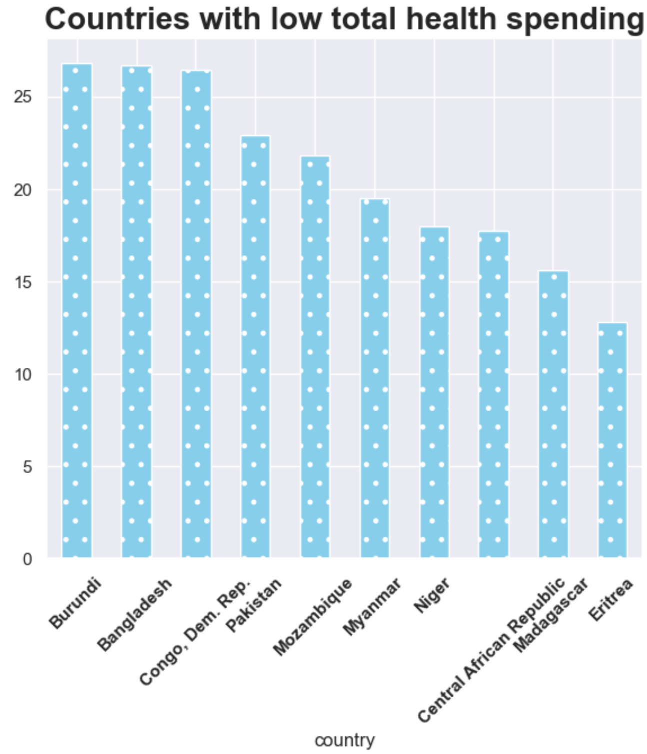
Out[172]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gc
159	United States	7.3000	6001.6000	8663.6000	7647.2000	49400	1.2200	78.7000	1.9300	484
145	Switzerland	4.5000	47744.0000	8579.0000	39761.8000	55500	0.3170	82.2000	1.5200	746
114	Norway	3.2000	34856.6000	8323.4400	25023.0000	62300	5.9500	81.0000	1.9500	878
91	Luxembourg	2.8000	183750.0000	8158.5000	149100.0000	91700	3.6200	81.3000	1.6300	1050
44	Denmark	4.1000	29290.0000	6612.0000	25288.0000	44000	3.2200	79.5000	1.8700	580
110	Netherlands	4.5000	36216.0000	5985.7000	31990.8000	45500	0.8480	80.7000	1.7900	503
29	Canada	5.6000	13793.4000	5356.2000	14694.0000	40700	2.8700	81.3000	1.6300	474
8	Austria	4.3000	24059.7000	5159.0000	22418.2000	43200	0.8730	80.5000	1.4400	469

In [173…

```
plt.figure(figsize=(15, 6))
He=df1.groupby('country').health.sum().sort_values(ascending= False)
plt.subplot(1,2,1)
He1=He.head(10).plot.bar(color='mediumorchid',hatch=".")
plt.title('Countries with high total health spending',fontweight="bold", size=20)
plt.xticks(rotation = 45,fontweight="bold")
plt.subplot(1,2,2)
He2=He.tail(10).plot.bar(color= 'skyblue',hatch='.')
plt.title('Countries with low total health spending',fontweight="bold", size=20)
plt.xticks(rotation = 45,fontweight="bold")
plt.show()
```





total FERTILITY of a country

In [174…

```
Fertility=df1.sort_values(by=['total_fer'], ascending= True).head(8)
Fertility
```

Out[174]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
133	Singapore	2.8000	93200.0000	1845.3600	81084.0000	72100	-0.0460	82.7000	1.1500	46600
138	South Korea	4.1000	10917.4000	1531.5300	10210.2000	30400	3.1600	80.1000	1.2300	22100
67	Hungary	6.0000	10715.8000	960.2300	10021.5000	22300	2.3300	74.5000	1.2500	13100
102	Moldova	17.2000	638.9600	190.7100	1279.5500	3910	11.1000	69.7000	1.2700	1630
20	Bosnia and Herzegovina	6.9000	1369.1700	511.7100	2364.9300	9720	1.4000	76.8000	1.3100	4610
98	Malta	6.8000	32283.0000	1825.1500	32494.0000	28300	3.8300	80.3000	1.3600	21100
85	Latvia	7.8000	6068.1000	754.8400	6226.3000	18300	-0.8120	73.1000	1.3600	11300
139	Spain	3.8000	7828.5000	2928.7800	8227.6000	32500	0.1600	81.9000	1.3700	30700

In [175…

```
Fertility1=df1.sort_values(by=['total_fer'], ascending=False).head(8)
Fertility1
```

Loading [MathJax]/extensions/Safe.js

Out[175]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
112	Niger	123.0000	77.2560	17.9568	170.8680	814	2.5500	58.8000	7.4900	348
32	Chad	150.0000	330.0960	40.6341	390.1950	1930	6.3900	56.5000	6.5900	897
97	Mali	137.0000	161.4240	35.2584	248.5080	1870	4.3700	59.5000	6.5500	708
37	Congo, Dem. Rep.	116.0000	137.2740	26.4194	165.6640	609	20.8000	57.5000	6.5400	334
26	Burundi	93.6000	20.6052	26.7960	90.5520	764	12.3000	57.7000	6.2600	231
149	Timor-Leste	62.6000	79.2000	328.3200	1000.8000	1850	26.5000	71.1000	6.2300	3600
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900	22.4000	60.1000	6.1600	3530
155	Uganda	81.0000	101.7450	53.6095	170.1700	1540	10.6000	56.8000	6.1500	595

In [176…

```
df1.total_fer.max()
```

Out[176]:

```
7.49
```

In [177…

```
df1.life_expec.max()
```

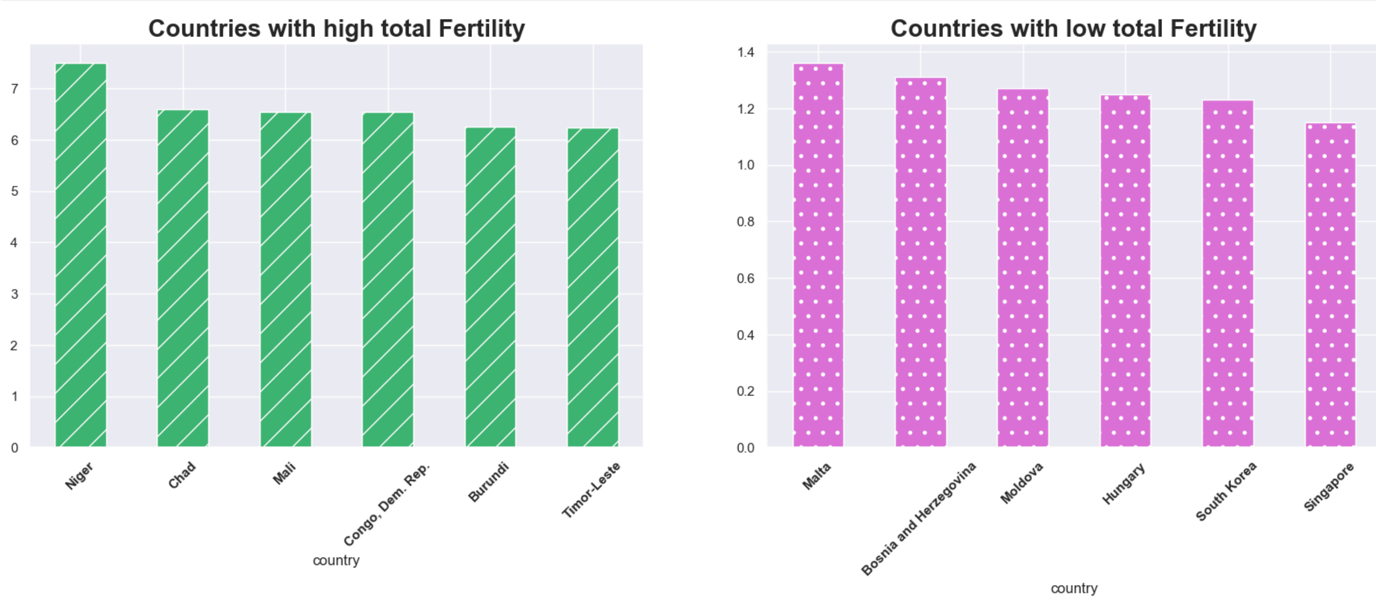
Out[177]:

```
82.8
```

countries with high and low FERTILITY

In [178…

```
plt.figure(figsize=(20, 6))
Fe=df1.groupby('country').total_fer.sum().sort_values(ascending= False)
plt.subplot(1,2,1)
Fe1=Fe.head(6).plot.bar(color='mediumseagreen',hatch="/")
plt.title('Countries with high total Fertility',fontweight="bold", size=20)
plt.xticks(rotation = 45,fontweight="bold")
plt.subplot(1,2,2)
Fe2=Fe.tail(6).plot.bar(color= 'orchid',hatch='.')
plt.title('Countries with low total Fertility',fontweight="bold", size=20)
plt.xticks(rotation = 45,fontweight="bold")
plt.show()
```



INFLATION

Loading [MathJax]/extensions/Safe.js

In [179…

```
In=df1.sort_values(by=['inflation'], ascending=False).head(8)
In
```

Out[179]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
113	Nigeria	130.0000	589.4900	118.1310	405.4200	5150	104.0000	60.5000	5.8400	2330
163	Venezuela	17.1000	3847.5000	662.8500	2376.0000	16500	45.9000	75.4000	2.4700	13500
103	Mongolia	26.1000	1237.5500	144.1600	1502.5500	7710	39.2000	66.2000	2.6400	2650
149	Timor-Leste	62.6000	79.2000	328.3200	1000.8000	1850	26.5000	71.1000	6.2300	3600
49	Equatorial Guinea	111.0000	14671.8000	766.0800	10071.9000	33700	24.9000	60.9000	5.2100	17100
165	Yemen	56.3000	393.0000	67.8580	450.6400	4480	23.6000	67.5000	4.6700	1310
140	Sri Lanka	11.2000	550.7600	82.6140	753.0800	8560	22.8000	74.4000	2.2000	2810
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900	22.4000	60.1000	6.1600	3530

In [180…

```
In=df1.sort_values(by=['inflation'], ascending=True).head(8)
In
```

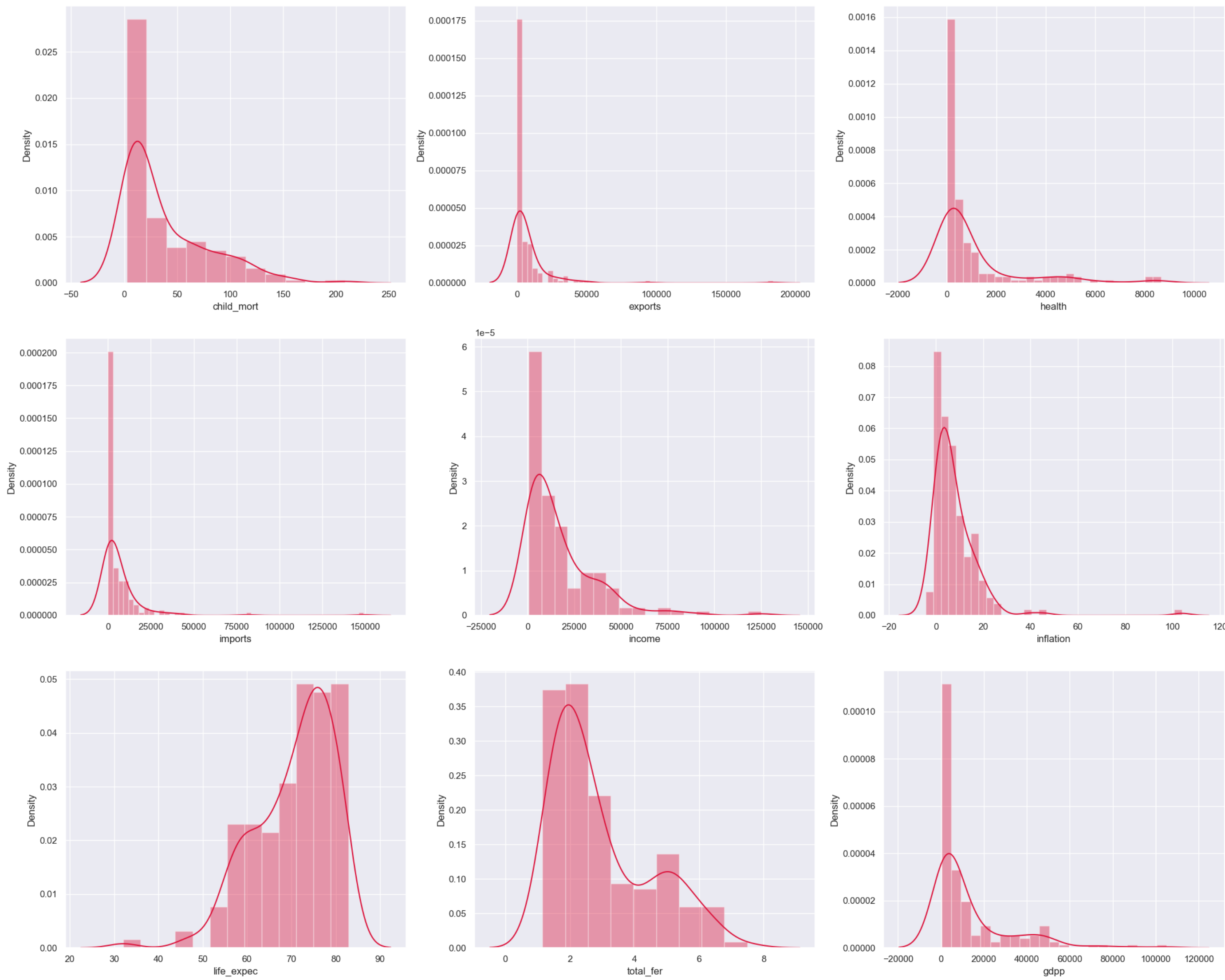
Out[180]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
131	Seychelles	14.4000	10130.4000	367.2000	11664.0000	20400	-4.2100	73.4000	2.1700	10800
73	Ireland	4.2000	50161.0000	4475.5300	42125.5000	45700	-3.2200	80.4000	2.0500	48700
77	Japan	3.2000	6675.0000	4223.0500	6052.0000	35800	-1.9000	82.8000	1.3900	44500
43	Czech Republic	3.4000	13068.0000	1560.2400	12454.2000	28300	-1.4300	77.5000	1.5100	19800
135	Slovenia	3.2000	15046.2000	2201.9400	14718.6000	28700	-0.9870	79.5000	1.5700	23400
85	Latvia	7.8000	6068.1000	754.8400	6226.3000	18300	-0.8120	73.1000	1.3600	11300
10	Bahamas	13.8000	9800.0000	2209.2000	12236.0000	22900	-0.3930	73.8000	1.8600	28000
133	Singapore	2.8000	93200.0000	1845.3600	81084.0000	72100	-0.0460	82.7000	1.1500	46600

DISPLOT

In [181…

```
plt.figure(figsize = (20,16))
feature = df1.columns[1:]
for i in enumerate(feature):
    plt.subplot(3,3, i[0]+1)
    sns.distplot(df1[i[1]], color='crimson')
    plt.subplots_adjust(right=1.1)
    plt.subplots_adjust(top=1.1)
```



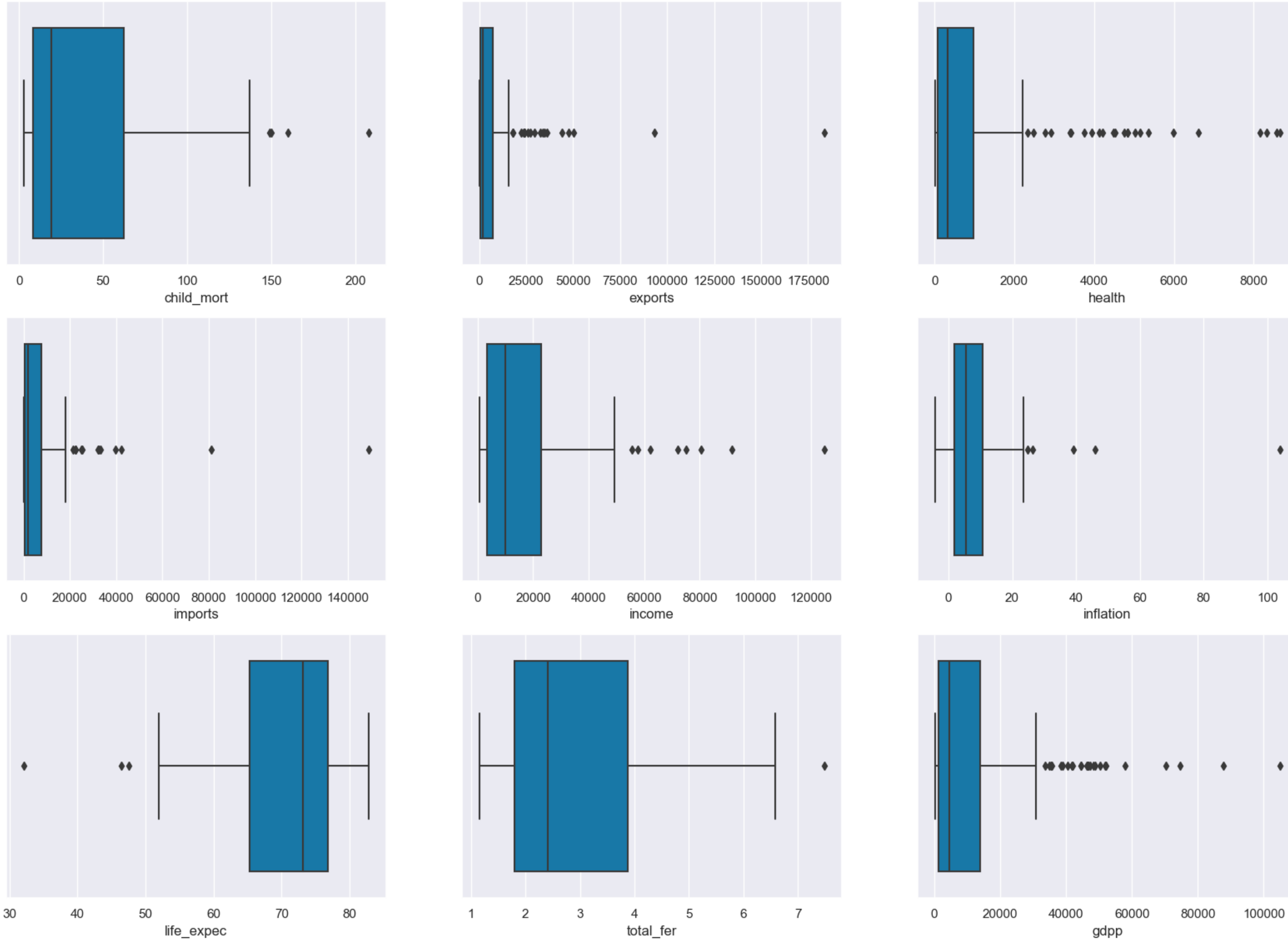
it is basicall used for unvarient set of observation and visualize it through histogram

only one observation.hence we choose one particular column of dataset

checking OUTLIERS

In [182…

```
plt.figure(figsize = (20,14))
feature = df1.columns[1:]
for i in enumerate(feature):
    plt.subplot(3,3, i[0]+1)
    sns.boxplot(df1[i[1]],palette='winter')
```



In [183…

```
#There are outliers in the data. we need to treat with The process of clustering that is
#outliers treatment
df1.describe()
```

Out[183]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gd
count	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0C
mean	38.2701	7420.6188	1056.7332	6588.3521	17144.6886	7.7818	70.5557	2.9480	12964.15
std	40.3289	17973.8858	1801.4089	14710.8104	19278.0677	10.5707	8.8932	1.5138	18328.7C
min	2.6000	1.0769	12.8212	0.6511	609.0000	-4.2100	32.1000	1.1500	231.0C
25%	8.2500	447.1400	78.5355	640.2150	3355.0000	1.8100	65.3000	1.7950	1330.0C
50%	19.3000	1777.4400	321.8860	2045.5800	9960.0000	5.3900	73.1000	2.4100	4660.0C
75%	62.1000	7278.0000	976.9400	7719.6000	22800.0000	10.7500	76.8000	3.8800	14050.0C
max	208.0000	183750.0000	8663.6000	149100.0000	125000.0000	104.0000	82.8000	7.4900	105000.0C

Capping

In [184…

```
q2 = df1['imports'].quantile(0.99)
q3 = df1['health'].quantile(0.99)
q4 = df1['income'].quantile(0.99)
q5 = df1['inflation'].quantile(0.99)
q6 = df1['life_expec'].quantile(0.99)
q7 = df1['total_fer'].quantile(0.99)
q8 = df1['gdpp'].quantile(0.99)
```

```
df1['imports'][df1['imports']>= q2] = q2
df1['health'][df1['health']>= q3] = q3
df1['income'][df1['income']>= q4] = q4
df1['inflation'][df1['inflation']>= q5] = q5
df1['life_expec'][df1['life_expec']>= q6] = q6
df1['total_fer'][df1['total_fer']>= q7] = q7
df1['gdpp'][df1['gdpp']>= q8] = q8
```

outliers after treatment

In [185…

```
df1.describe()
```

Out[185]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
count	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000	167.0000
mean	38.2701	7420.6188	1054.2066	5873.1352	16857.5509	7.3810	70.5511	2.9423	12756.8263
std	40.3289	17973.8858	1790.8453	9422.7009	17957.0129	7.7932	8.8870	1.4983	17430.2089
min	2.6000	1.0769	12.8212	0.6511	609.0000	-4.2100	32.1000	1.1500	231.0000
25%	8.2500	447.1400	78.5355	640.2150	3355.0000	1.8100	65.3000	1.7950	1330.0000
50%	19.3000	1777.4400	321.8860	2045.5800	9960.0000	5.3900	73.1000	2.4100	4660.0000
75%	62.1000	7278.0000	976.9400	7719.6000	22800.0000	10.7500	76.8000	3.8800	14050.0000
max	208.0000	183750.0000	8410.3304	55371.3900	84374.0000	41.4780	82.3700	6.5636	79088.0000

CLUSTERING of data

In [186…

```
from sklearn.neighbors import NearestNeighbors
from random import sample
from numpy.random import uniform
import numpy as np
from math import isnan

def hopkins(X):
    d = X.shape[1]
    n = len(X)
    m = int(0.1 * n)
    nbrs = NearestNeighbors(n_neighbors=1).fit(X.values)

    rand_X = sample(range(0, n, 1), m)
    ujd = []
    wjd = []
    for j in range(0, m):
        u_dist, _ = nbrs.kneighbors(uniform(np.amin(X,axis=0),np.amax(X,axis=0),d).resha
        ujd.append(u_dist[0][1])
        w_dist, _ = nbrs.kneighbors(X.iloc[rand_X[j]].values.reshape(1, -1), 2, return_d
        wjd.append(w_dist[0][1])

    H = sum(ujd) / (sum(ujd) + sum(wjd))
    if isnan(H):
        print(ujd, wjd)
        H = 0

    return H
```

In [187

```
df1.head(100)
```

Out[187]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	
0	Afghanistan	90.2000	55.3000	41.9174	248.2970	1610.0000	9.4400	56.2000	5.8200	55
1	Albania	16.6000	1145.2000	267.8950	1987.7400	9930.0000	4.4900	76.3000	1.6500	409
2	Algeria	27.3000	1712.6400	185.9820	1400.4400	12900.0000	16.1000	76.5000	2.8900	446
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900.0000	22.4000	60.1000	6.1600	353
4	Antigua and Barbuda	10.3000	5551.0000	735.6600	7185.8000	19100.0000	1.4400	76.8000	2.1300	1220
...	...	...	...	...	...	...	...	...	...	
95	Malaysia	7.9000	7881.8300	398.1730	6439.7000	21100.0000	7.2700	74.5000	2.1500	907
96	Maldives	13.2000	5509.6000	449.4300	4643.4000	10500.0000	2.8800	77.9000	2.2300	710
97	Mali	137.0000	161.4240	35.2584	248.5080	1870.0000	4.3700	59.5000	6.5500	70
98	Malta	6.8000	32283.0000	1825.1500	32494.0000	28300.0000	3.8300	80.3000	1.3600	2110
99	Mauritania	97.4000	608.4000	52.9200	734.4000	3320.0000	18.9000	68.2000	4.9800	120

100 rows × 10 columns

In [188…

```
hopkins(df1.drop('country', axis = 1))
```

Out[188]:

```
0.9139792624010575
```

SCALING

In [189…

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
data1 = scaler.fit_transform(df1.drop('country', axis = 1))
data1
```

Out[189]:

```
array([[ 1.29153238, -0.4110113 , -0.56695778, ..., -1.61970522,
         1.92639646, -0.70225949],
       [-0.5389489 , -0.35019096, -0.4403934 , ...,  0.64883094,
        -0.86505432, -0.49872564],
       [-0.27283273, -0.31852577, -0.48627082, ...,  0.67140344,
        -0.03498262, -0.47743428],
       ...,
       [-0.37231541, -0.36146329, -0.54024972, ...,  0.28767096,
        -0.66423052, -0.65869853],
       [ 0.44841668, -0.39216643, -0.55242911, ..., -0.34435902,
         1.15657191, -0.65869853],
       [ 1.11495062, -0.38395214, -0.54227159, ..., -2.09372771,
         1.64524315, -0.6500669 ]])
```

In [190…

```
data1 = pd.DataFrame(data1, columns = df1.columns[1:])
data1.head()
```

Loading [MathJax]/extensions/Safe.js

Out[190]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	1.2915	-0.4110	-0.5670	-0.5987	-0.8517	0.2650	-1.6197	1.9264	-0.7023
1	-0.5389	-0.3502	-0.4404	-0.4136	-0.3869	-0.3721	0.6488	-0.8651	-0.4987
2	-0.2728	-0.3185	-0.4863	-0.4761	-0.2211	1.1222	0.6714	-0.0350	-0.4774
3	2.0078	-0.2914	-0.5341	-0.4640	-0.6120	1.9330	-1.1795	2.1540	-0.5310
4	-0.6956	-0.1043	-0.1784	0.1397	0.1253	-0.7646	0.7053	-0.5437	-0.0320

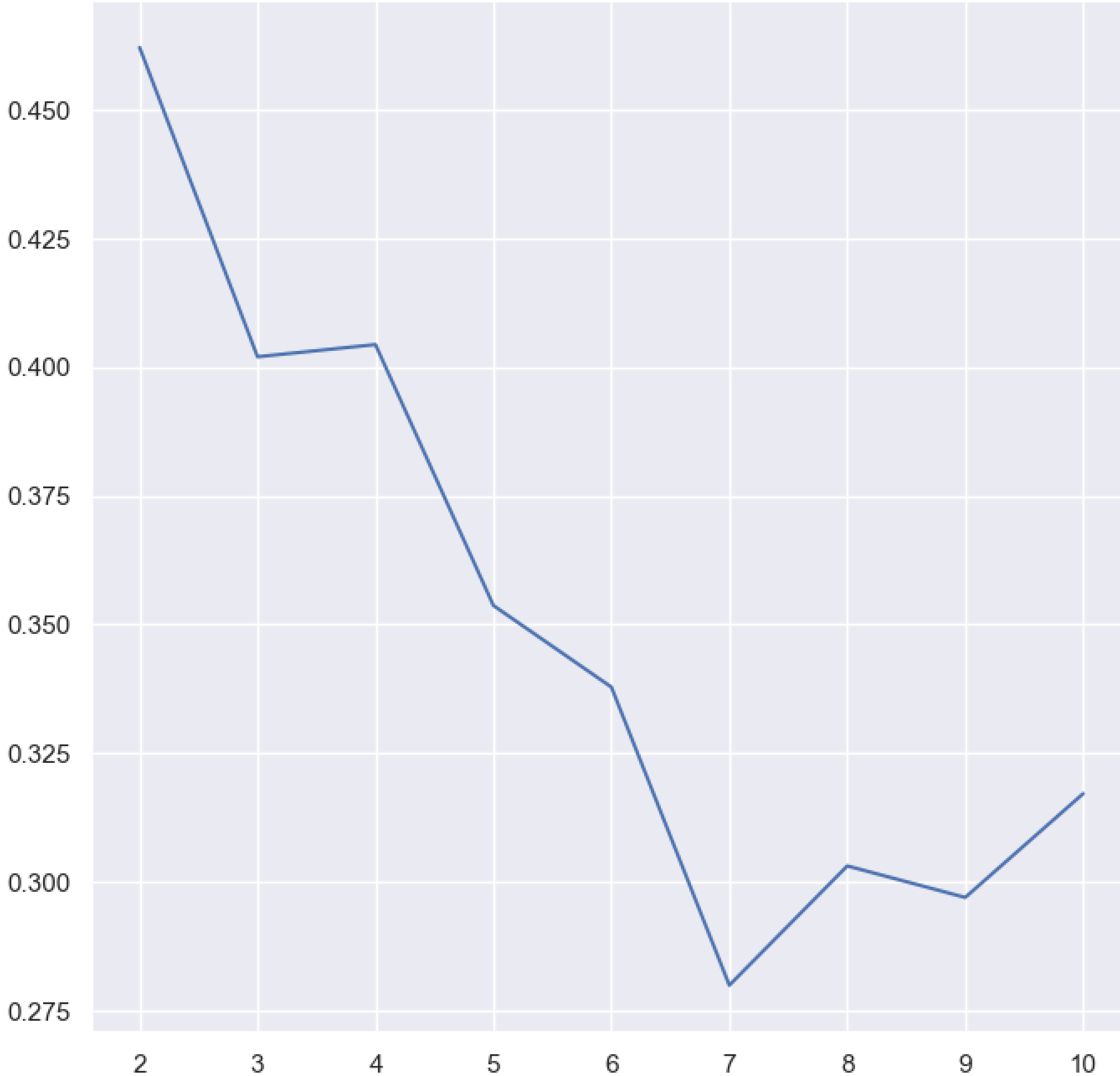
K-Mean Clustering

Silhouette score

In [191…

```
from sklearn.metrics import silhouette_score
ss = []
for k in range(2, 11):
    kmean = KMeans(n_clusters = k).fit(data1)
    ss.append([k, silhouette_score(data1, kmean.labels_)])
temp = pd.DataFrame(ss)
plt.plot(temp[0], temp[1])

plt.show()
```

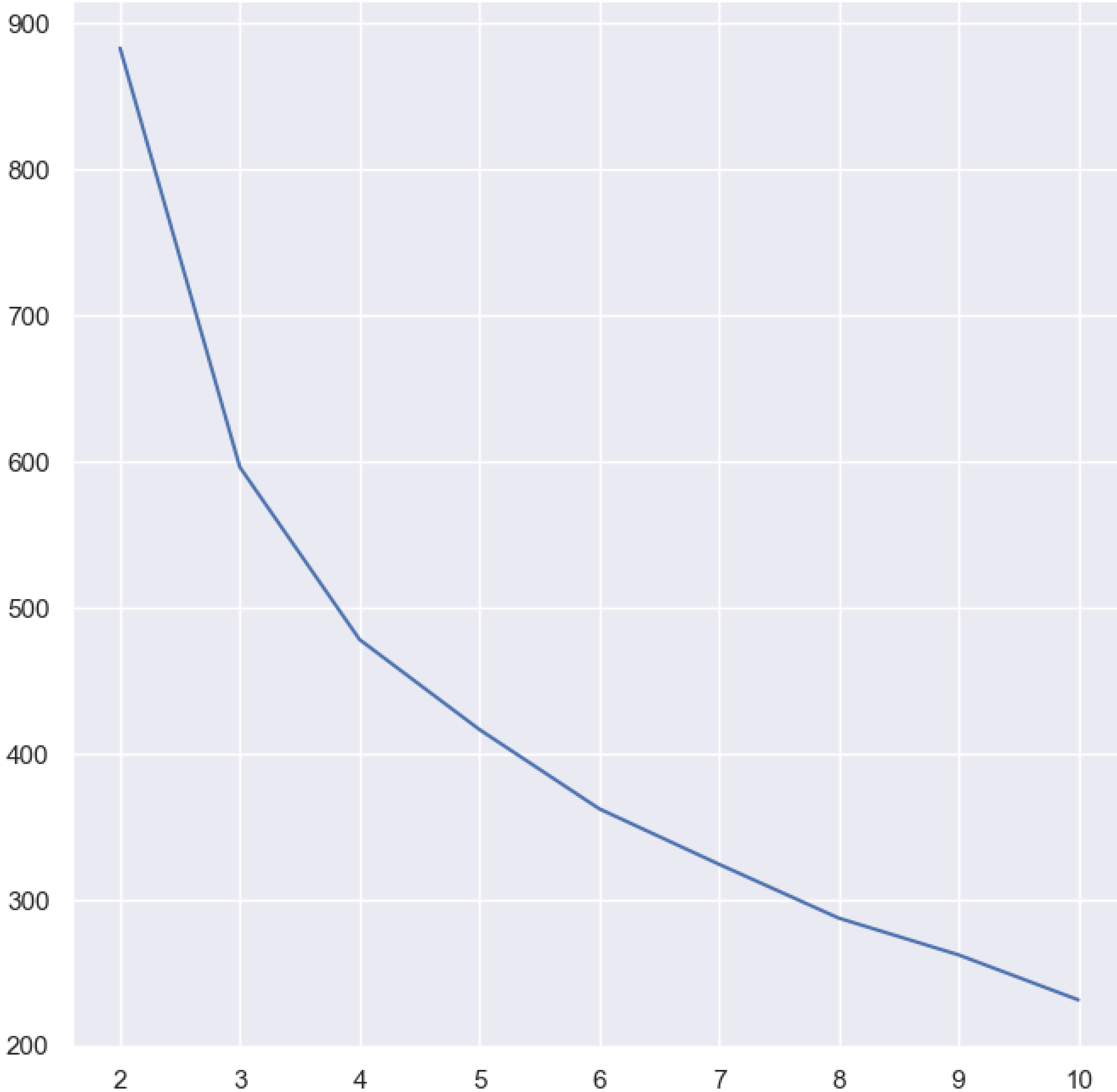


Elbow curve

In [192…

```
ssd = []
for k in range(2, 11):
    kmean = KMeans(n_clusters = k).fit(data1)
    ssd.append([k, kmean.inertia_])

temp = pd.DataFrame(ssd)
plt.plot(temp[0], temp[1])
plt.show()
```



K=3

Final Kmean Clustering

In [193…

```
kmean = KMeans(n_clusters = 3, random_state = 50)
kmean.fit(data1)
```

Out[193]:

```
KMeans(n_clusters=3, random_state=50)
```

In [194…

```
df1_kmean = df1.copy()
```

In [195…

```
label  = pd.DataFrame(kmean.labels_, columns= ['label'])
label.head()
```

Out[195]:

	label
0	2
1	1
2	1
3	2
4	1

In [196…

```
df1_kmean = pd.concat([df1_kmean, label], axis =1)
df1_kmean.head(100)
```

Out[196]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	
0	Afghanistan	90.2000	55.3000	41.9174	248.2970	1610.0000	9.4400	56.2000	5.8200	55
1	Albania	16.6000	1145.2000	267.8950	1987.7400	9930.0000	4.4900	76.3000	1.6500	409
2	Algeria	27.3000	1712.6400	185.9820	1400.4400	12900.0000	16.1000	76.5000	2.8900	446
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900.0000	22.4000	60.1000	6.1600	353
4	Antigua and Barbuda	10.3000	5551.0000	735.6600	7185.8000	19100.0000	1.4400	76.8000	2.1300	1220
...	...	...	...	...	...	...	...	...	...	
95	Malaysia	7.9000	7881.8300	398.1730	6439.7000	21100.0000	7.2700	74.5000	2.1500	907
96	Maldives	13.2000	5509.6000	449.4300	4643.4000	10500.0000	2.8800	77.9000	2.2300	710
97	Mali	137.0000	161.4240	35.2584	248.5080	1870.0000	4.3700	59.5000	6.5500	70
98	Malta	6.8000	32283.0000	1825.1500	32494.0000	28300.0000	3.8300	80.3000	1.3600	2110
99	Mauritania	97.4000	608.4000	52.9200	734.4000	3320.0000	18.9000	68.2000	4.9800	120

100 rows × 11 columns

In [197…

```
df1_kmean.label.value_counts()
```

Out[197]:

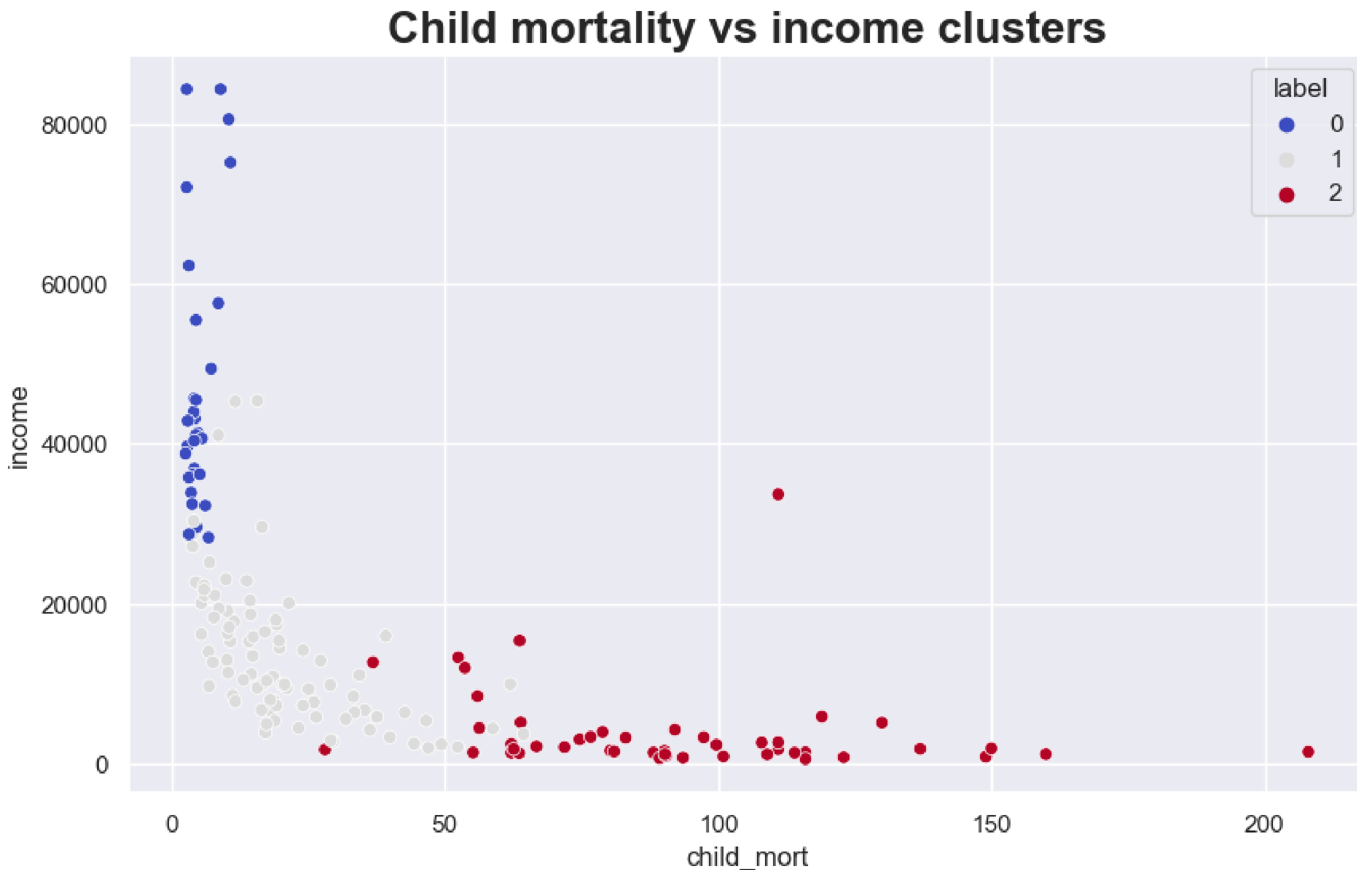
```
1    89
2    48
0    30
Name: label, dtype: int64
```

Plot the cluster

In [198…

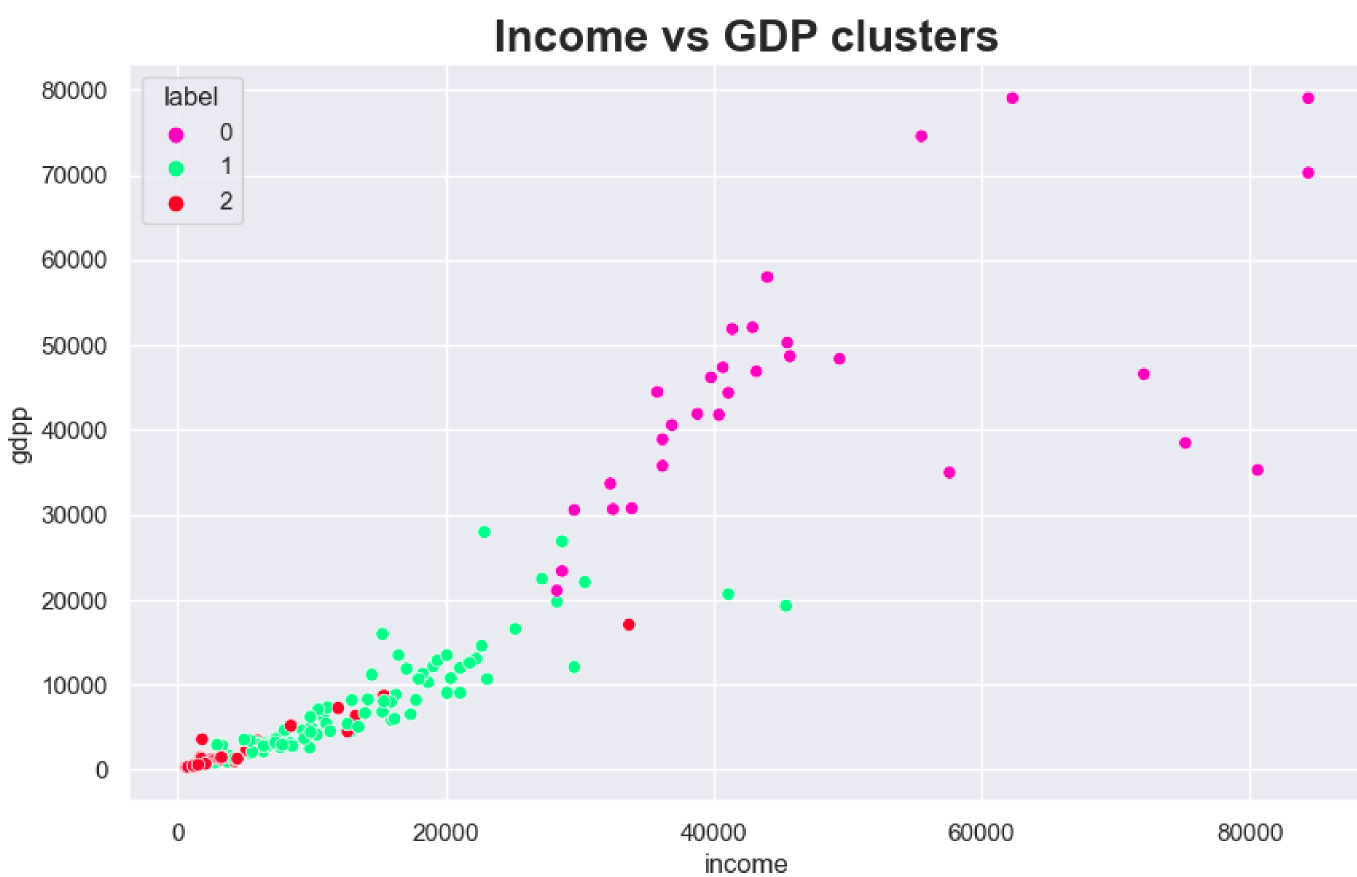
```
plt.figure(figsize = (10,6))
sns.scatterplot(x = 'child_mort', y = 'income', hue = 'label', data = df1_kmean, palette
plt.title('Child mortality vs income clusters',fontweight="bold", size=20)

plt.show()
```



In [199…

```
plt.figure(figsize = (10,6))
sns.scatterplot(x = 'income', y = 'gdpp', hue = 'label', data = df1_kmean, palette = 'gi
plt.title('Income vs GDP clusters',fontweight="bold", size=20)
plt.show()
```



In [200…

```
plt.figure(figsize = (10,6))
sns.scatterplot(x = 'child_mort', y = 'gdpp', hue = 'label', data= df1_kmean, palette =
```

```
plt.title('Child mortality vs GDP clusters',fontweight="bold", size=20 )
plt.show()
```

cluster profiling

In [201…

```
df1=df1_kmean.drop('country', axis = 1).groupby('label').mean()
df1
```

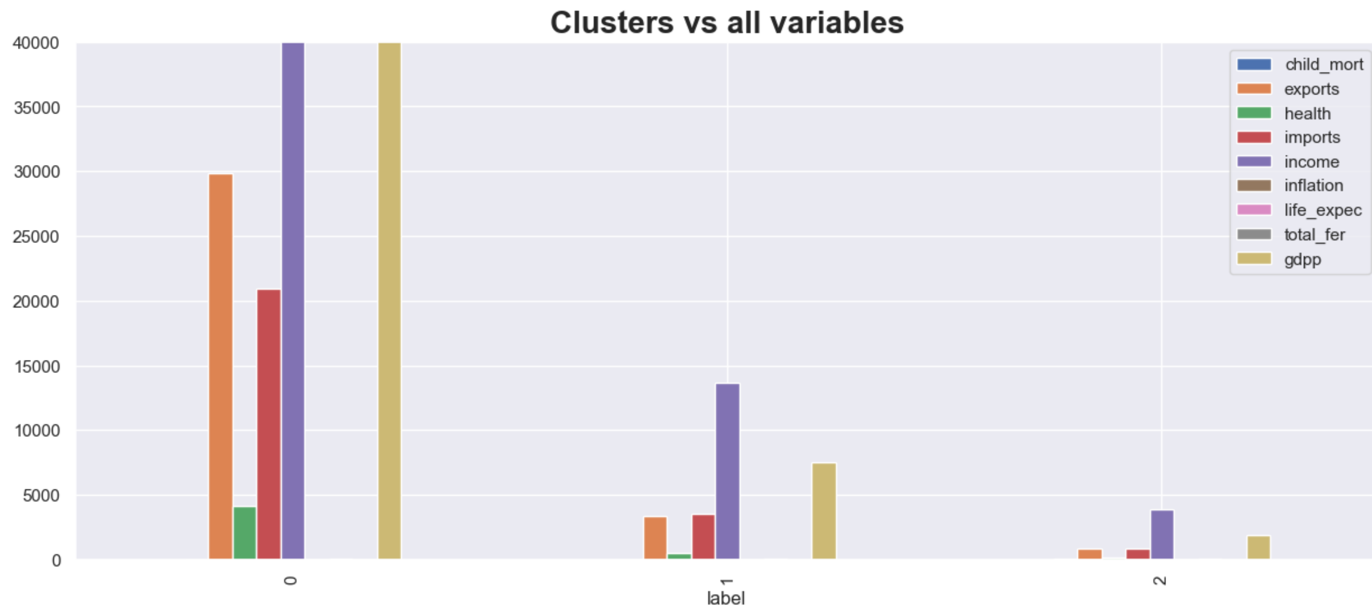
Out[201]:

label	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	4.9700	29827.5633	4175.8450	20941.7193	47178.2667	2.8398	80.4847	1.7967	45552.5333
1	20.7270	3395.7461	508.6035	3515.3328	13626.8539	7.1710	73.3034	2.2336	7552.4944
2	91.6104	879.0635	114.8218	827.0288	3897.3542	10.6086	59.2396	4.9722	1909.2083

In [202…

```
df1_kmean.drop('country', axis = 1).groupby('label').mean().plot.bar(figsize=(15,6))
plt.ylim([0,40000])
plt.title("Clusters vs all variables",fontweight="bold", size=20)

plt.show()
```



In [203…

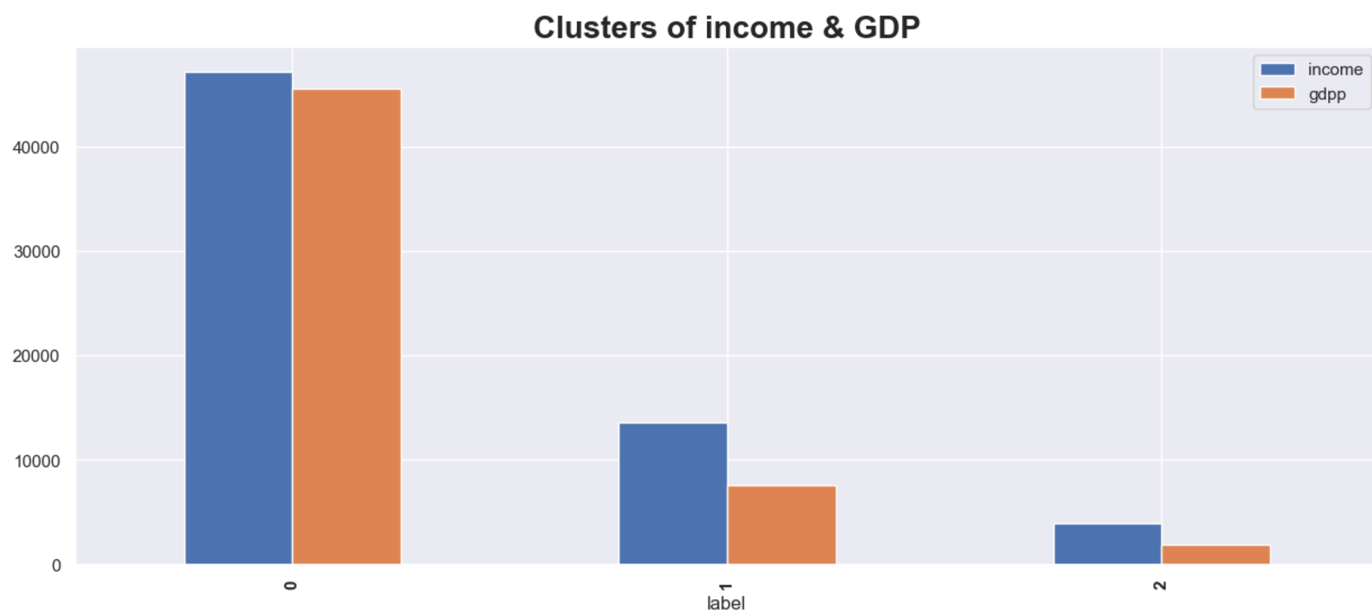
```
df1_kmean.columns
```

Out[203]:

```
Index(['country', 'child_mort', 'exports', 'health', 'imports', 'income',
       'inflation', 'life_expec', 'total_fer', 'gdpp', 'label'],
      dtype='object')
```

In [204…

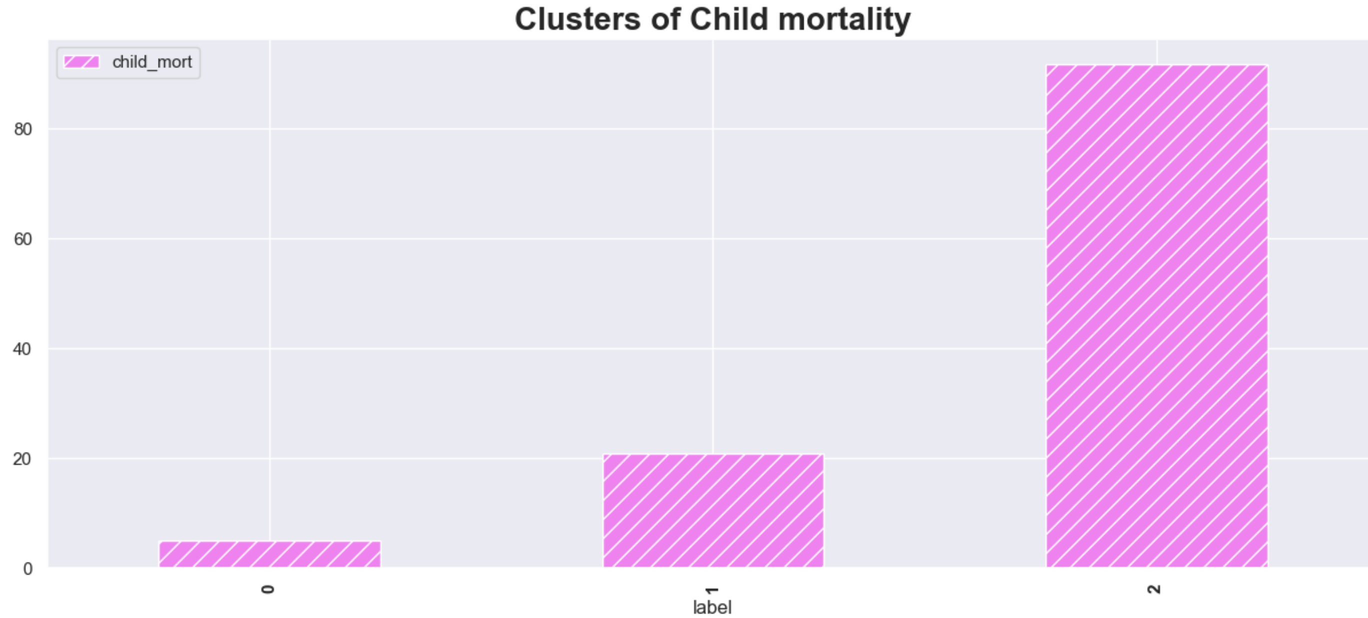
```
df1_kmean.drop(['country', 'exports', 'health', 'imports','child_mort',
       'inflation', 'life_expec', 'total_fer'],axis = 1).groupby('label').mean().plot.ba
plt.title('Clusters of income & GDP',fontweight="bold", size=20)
plt.xticks(fontweight='bold')
plt.show()
```



In [205…

```
df1_kmean.drop(['country', 'exports', 'health', 'imports','income', 'gdpp',
       'inflation', 'life_expec', 'total_fer'],axis = 1).groupby('label').mean().plot.ba
plt.title('Clusters of Child mortality', fontweight="bold", size=20)
plt.xticks(fontweight='bold')
plt.show()
```

Loading [MathJax]/extensions/Safe.js



From cluster profiling in K- means clustering we can see that :

1. Cluster 0 is having the High income, High GDP and very Low child mortality
2. Cluster 2 is having very Low income, very Low GDP but High child mortality
3. Cluster 1 is having low income, GDP and less child mortality
4. We saw in cluster profiling that cluster 2 is having low income, low GDP and High Child Mortality
5. So we can say that countries under cluster 2 are in need of aid. Lets see the countries

In [206…

```
Kmean=df1_kmean[df1_kmean['label'] == 2]
Kmean.head()
```

Out[206]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gd
0	Afghanistan	90.2000	55.3000	41.9174	248.2970	1610.0000	9.4400	56.2000	5.8200	553.00
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900.0000	22.4000	60.1000	6.1600	3530.00
17	Benin	111.0000	180.4040	31.0780	281.9760	1820.0000	0.8850	61.8000	5.3600	758.00
21	Botswana	52.5000	2768.6000	527.0500	3257.5500	13300.0000	8.9200	57.1000	2.8800	6350.00
25	Burkina Faso	116.0000	110.4000	38.7550	170.2000	1430.0000	6.8100	57.9000	5.8700	575.00

Countries that we have to focus more on:

In [207…

```
K=Kmean[['country']]
K= K.reset_index(drop=True)
K
```

Out[207]:

	country
0	Afghanistan
1	Angola
2	Benin
3	Botswana
4	Burkina Faso
5	Burundi
6	Cameroon
7	Central African Republic
8	Chad
9	Comoros
10	Congo, Dem. Rep.
11	Congo, Rep.
12	Cote d'Ivoire
13	Equatorial Guinea
14	Eritrea
15	Gabon
16	Gambia
17	Ghana
18	Guinea
19	Guinea-Bissau
20	Haiti
21	Iraq
22	Kenya
23	Kiribati
24	Lao
25	Lesotho
26	Liberia
27	Madagascar
28	Malawi
29	Mali
30	Mauritania
31	Mozambique
32	Namibia
33	Niger
34	Nigeria
35	Pakistan
36	Rwanda
37	Senegal
38	Sierra Leone

	country
39	Solomon Islands
40	South Africa
41	Sudan
42	Tanzania
43	Timor-Leste
44	Togo
45	Uganda
46	Yemen
47	Zambia

CLUSTERING that is based on the high child mortality,low income and GDP

In [208…

```
final=df1_kmean[df1_kmean['label'] == 2].sort_values(by = ['child_mort','income', 'gdpp'
final.head(10)
```

Out[208]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
66	Haiti	208.0000	101.2860	45.7442	428.3140	1500.0000	5.4500	32.1000	3.3300	662.0000
132	Sierra Leone	160.0000	67.0320	52.2690	137.6550	1220.0000	17.2000	55.0000	5.2000	399.0000
32	Chad	150.0000	330.0960	40.6341	390.1950	1930.0000	6.3900	56.5000	6.5636	897.0000
31	Central African Republic	149.0000	52.6280	17.7508	118.1900	888.0000	2.0100	47.5000	5.2100	446.0000
97	Mali	137.0000	161.4240	35.2584	248.5080	1870.0000	4.3700	59.5000	6.5500	708.0000
113	Nigeria	130.0000	589.4900	118.1310	405.4200	5150.0000	41.4780	60.5000	5.8400	2330.0000
112	Niger	123.0000	77.2560	17.9568	170.8680	814.0000	2.5500	58.8000	6.5636	348.0000
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900.0000	22.4000	60.1000	6.1600	3530.0000
37	Congo, Dem. Rep.	116.0000	137.2740	26.4194	165.6640	609.0000	20.8000	57.5000	6.5400	334.0000
25	Burkina Faso	116.0000	110.4000	38.7550	170.2000	1430.0000	6.8100	57.9000	5.8700	575.0000

In [209…

```
print("Top 10 countries which are in direst need of aid" )
f=final[['country']].head(10)
df1_r = f.reset_index(drop=True)
df1_r
```

```
Top 10 countries which are in direst need of aid
```

Out[209]:

	country
0	Haiti
1	Sierra Leone
2	Chad
3	Central African Republic
4	Mali
5	Nigeria
6	Niger
7	Angola
8	Congo, Dem. Rep.
9	Burkina Faso

HIRARCHICAL CLUSTERING

In [210…

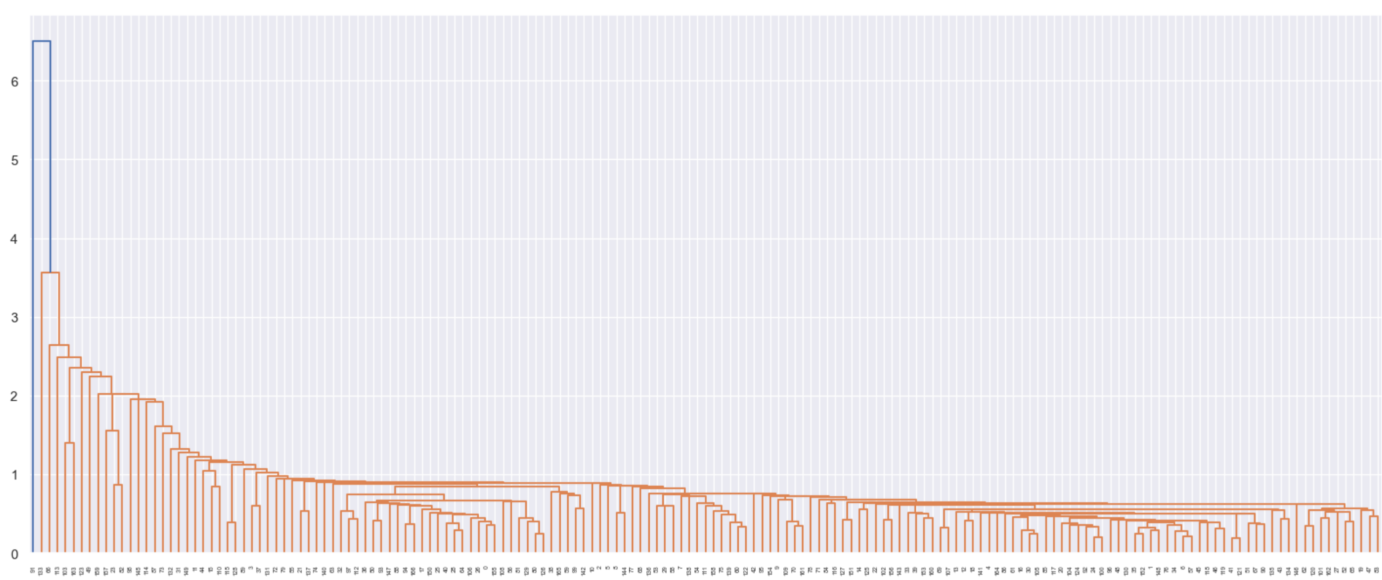
```
data1.head()
```

Out[210]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
0	1.2915	-0.4110	-0.5670	-0.5987	-0.8517	0.2650	-1.6197	1.9264	-0.7023
1	-0.5389	-0.3502	-0.4404	-0.4136	-0.3869	-0.3721	0.6488	-0.8651	-0.4987
2	-0.2728	-0.3185	-0.4863	-0.4761	-0.2211	1.1222	0.6714	-0.0350	-0.4774
3	2.0078	-0.2914	-0.5341	-0.4640	-0.6120	1.9330	-1.1795	2.1540	-0.5310
4	-0.6956	-0.1043	-0.1784	0.1397	0.1253	-0.7646	0.7053	-0.5437	-0.0320

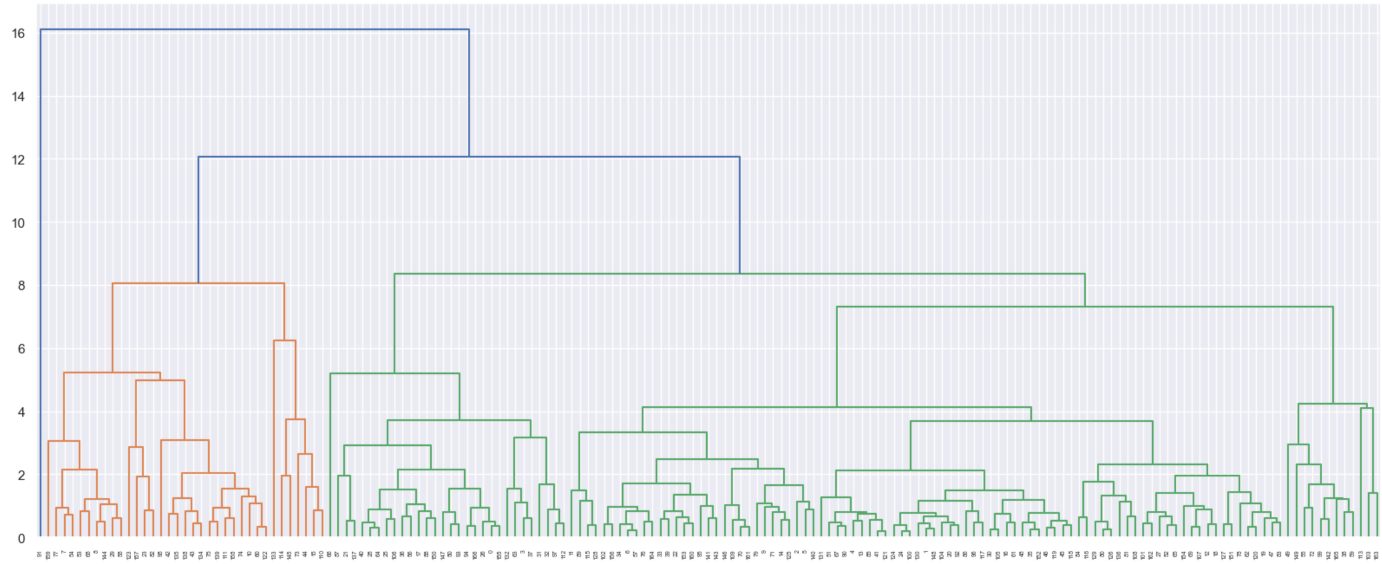
In [211…

```
plt.figure(figsize = (20,8))
mergings = linkage(data1, method="single", metric='euclidean')
dendrogram(mergings)
plt.show()
```



In [212…

```
plt.figure(figsize = (20,8))
mergings = linkage(data1, method="complete", metric='euclidean')
dendrogram(mergings)
plt.show()
```



Now we got the clear dendrogram and its easier to analyse the clusters. Lets consider a threshold value of 10. Draw the horizontal line at that height. It cuts 3 vertical lines, all of which represent a cluster. So we have 3 clusters now

clusters

In [213…
```
cluster_labels = cut_tree(mergings, n_clusters=3).reshape(-1, )
cluster_labels
```

Out[213]:
```
array([0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0])
```

In [214…
```
data1['cluster_labels'] = cluster_labels
data1.head()
```

Out[214]:

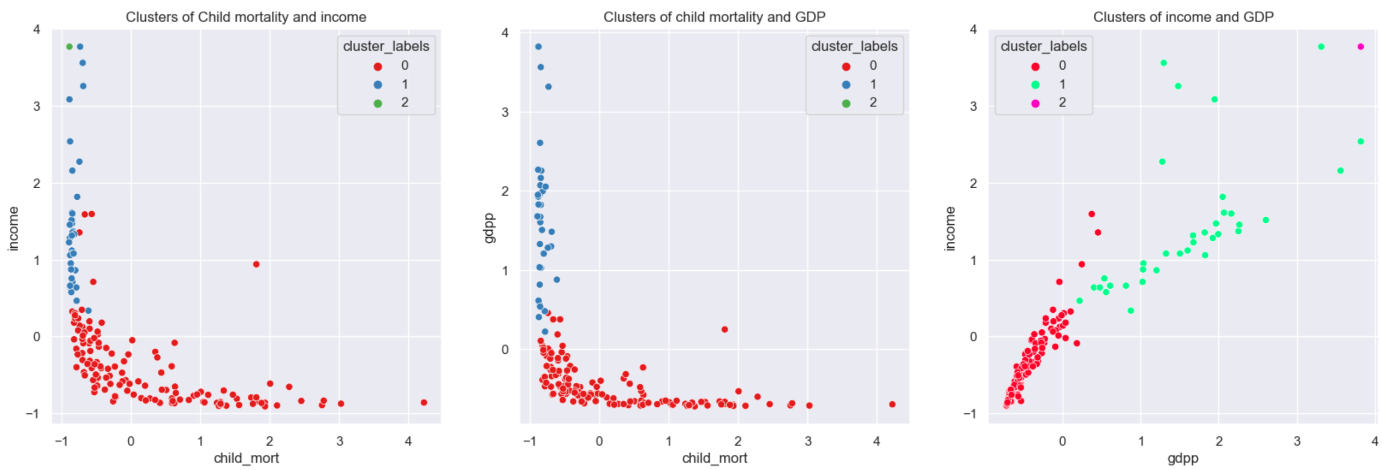
	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	cluster_labels
0	1.2915	-0.4110	-0.5670	-0.5987	-0.8517	0.2650	-1.6197	1.9264	-0.7023	0
1	-0.5389	-0.3502	-0.4404	-0.4136	-0.3869	-0.3721	0.6488	-0.8651	-0.4987	0
2	-0.2728	-0.3185	-0.4863	-0.4761	-0.2211	1.1222	0.6714	-0.0350	-0.4774	0
3	2.0078	-0.2914	-0.5341	-0.4640	-0.6120	1.9330	-1.1795	2.1540	-0.5310	0
4	-0.6956	-0.1043	-0.1784	0.1397	0.1253	-0.7646	0.7053	-0.5437	-0.0320	0

In [215…
```
data1.cluster_labels.value_counts()
```

Out[215]:
```
0    131
1     35
2      1
Name: cluster_labels, dtype: int64
```

In [216…
```
fig, axes = plt.subplots(1,3, figsize=(20,6))
plt.subplot(1,3,1)
sns.scatterplot(x='child_mort', y='income', hue='cluster_labels',data=data1, palette='Se
usters of Child mortality and income')
```

```
plt.subplot(1,3,2)
sns.scatterplot(x='child_mort', y='gdpp', hue='cluster_labels',data=data1, palette='Set1
plt.title('Clusters of child mortality and GDP')
plt.subplot(1,3,3)
sns.scatterplot(x='gdpp', y='income', hue='cluster_labels',data=data1,palette='gist_rain
plt.title('Clusters of income and GDP')
plt.show()
```

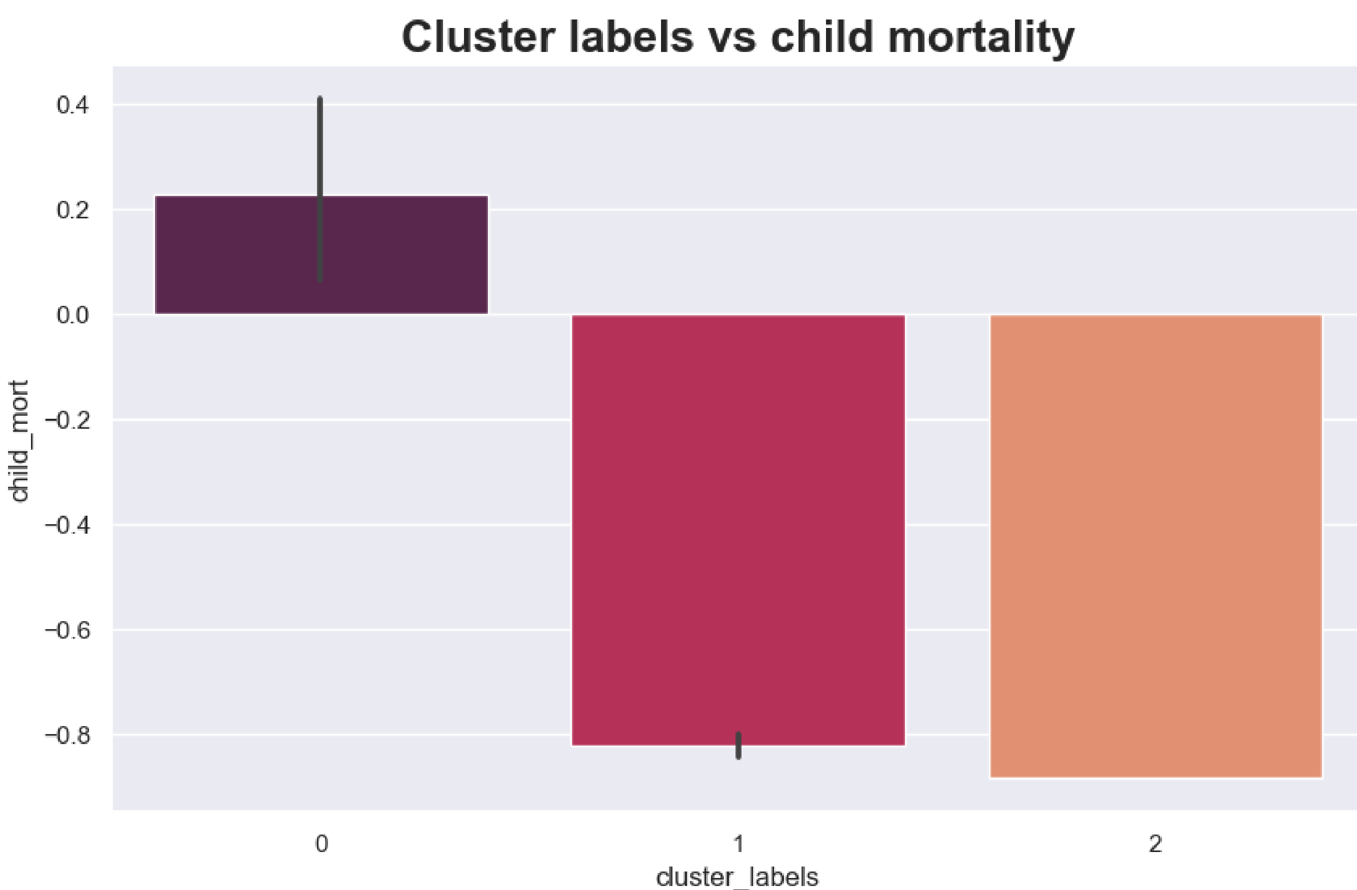


plots

In [217…

```
plt.figure(figsize = (10,6))
sns.barplot(x='cluster_labels', y='child_mort', data=data1,palette='rocket')
plt.title('Cluster labels vs child mortality',fontweight="bold", size=20)

plt.show()
```



In [218…

```
plt.figure(figsize = (10,6))
sns.barplot(x='cluster_labels', y='income', data=data1, palette='coolwarm')
```

```
plt.title('Cluster labels vs Income',fontweight="bold", size=20)

plt.show()
```

In [219…

```
plt.figure(figsize = (10,6))
sns.barplot(x='cluster_labels', y='gdpp', data=data1, palette='magma')
plt.title('Cluster labels vs GDP',fontweight="bold", size=20)

plt.show()
```

Loading [MathJax]/extensions/Safe.js


```
In [220…  data1[data1['cluster_labels'] == 0].head()
```

Out[220]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	cluster_labels
0	1.2915	-0.4110	-0.5670	-0.5987	-0.8517	0.2650	-1.6197	1.9264	-0.7023	0
1	-0.5389	-0.3502	-0.4404	-0.4136	-0.3869	-0.3721	0.6488	-0.8651	-0.4987	0
2	-0.2728	-0.3185	-0.4863	-0.4761	-0.2211	1.1222	0.6714	-0.0350	-0.4774	0
3	2.0078	-0.2914	-0.5341	-0.4640	-0.6120	1.9330	-1.1795	2.1540	-0.5310	0
4	-0.6956	-0.1043	-0.1784	0.1397	0.1253	-0.7646	0.7053	-0.5437	-0.0320	0

```
In [221…  data1=data1[data1['cluster_labels'] == 0].sort_values(by = ['child_mort','income', 'gdpp
          data1.head(10)
```

Out[221]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	cluster_labels
66	4.2213	-0.4084	-0.5648	-0.5796	-0.8578	-0.2485	-4.3397	0.2596	-0.6960	0
132	3.0275	-0.4104	-0.5612	-0.6105	-0.8735	1.2637	-1.7551	1.5114	-0.7111	0
32	2.7788	-0.3957	-0.5677	-0.5836	-0.8338	-0.1275	-1.5858	2.4242	-0.6825	0
31	2.7539	-0.4112	-0.5805	-0.6126	-0.8920	-0.6913	-2.6016	1.5181	-0.7084	0
97	2.4555	-0.4051	-0.5707	-0.5987	-0.8371	-0.3875	-1.2473	2.4151	-0.6933	0
113	2.2814	-0.3812	-0.5243	-0.5820	-0.6539	4.3884	-1.1344	1.9398	-0.6000	0
112	2.1073	-0.4098	-0.5804	-0.6070	-0.8961	-0.6218	-1.3263	2.4242	-0.7141	0
3	2.0078	-0.2914	-0.5341	-0.4640	-0.6120	1.9330	-1.1795	2.1540	-0.5310	0
37	1.9332	-0.4064	-0.5756	-0.6075	-0.9076	1.7271	-1.4730	2.4084	-0.7149	0
25	1.9332	-0.4079	-0.5687	-0.6071	-0.8617	-0.0735	-1.4278	1.9599	-0.7010	0

final Analysis

In [222…

```
Kmean.head()
```

Out[222]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gd
0	Afghanistan	90.2000	55.3000	41.9174	248.2970	1610.0000	9.4400	56.2000	5.8200	553.00
3	Angola	119.0000	2199.1900	100.6050	1514.3700	5900.0000	22.4000	60.1000	6.1600	3530.00
17	Benin	111.0000	180.4040	31.0780	281.9760	1820.0000	0.8850	61.8000	5.3600	758.00
21	Botswana	52.5000	2768.6000	527.0500	3257.5500	13300.0000	8.9200	57.1000	2.8800	6350.00
25	Burkina Faso	116.0000	110.4000	38.7550	170.2000	1430.0000	6.8100	57.9000	5.8700	575.00

In [223…

```
Kmean=Kmean.sort_values(by=['child_mort','income','gdpp'], ascending=[False,True,True])
Kmean.head()
```

Out[223]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	lab
66	Haiti	208.0000	101.2860	45.7442	428.3140	1500.0000	5.4500	32.1000	3.3300	662.0000	
132	Sierra Leone	160.0000	67.0320	52.2690	137.6550	1220.0000	17.2000	55.0000	5.2000	399.0000	
32	Chad	150.0000	330.0960	40.6341	390.1950	1930.0000	6.3900	56.5000	6.5636	897.0000	
31	Central African Republic	149.0000	52.6280	17.7508	118.1900	888.0000	2.0100	47.5000	5.2100	446.0000	
97	Mali	137.0000	161.4240	35.2584	248.5080	1870.0000	4.3700	59.5000	6.5500	708.0000	

In [224…

```
print('10 Countries based on high child mortality, low income and low GDP')
Kmean[['country']].head(10)
```

```
10 Countries based on high child mortality, low income and low GDP
```

Out[224]:

	country
66	Haiti
132	Sierra Leone
32	Chad
31	Central African Republic
97	Mali
113	Nigeria
112	Niger
3	Angola
37	Congo, Dem. Rep.
25	Burkina Faso

choosing the countries based on socio-economic and health factors

In [225…

```
Kmean.describe()
```

Out[225]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	
count	48.0000	48.0000	48.0000	48.0000	48.0000	48.0000	48.0000	48.0000	48.0000	48
mean	91.6104	879.0635	114.8218	827.0288	3897.3542	10.6086	59.2396	4.9722	1909.2083	2
std	34.3199	2252.4740	165.5183	1540.9819	5590.1686	8.5112	6.3849	0.9956	2925.9110	0
min	28.1000	20.6052	12.8212	90.5520	609.0000	0.8850	32.1000	2.5900	231.0000	2
25%	63.6750	102.8738	34.0059	193.3195	1390.0000	4.0800	56.7250	4.4750	551.5000	2
50%	89.7500	196.2600	51.6135	339.3060	1860.0000	8.8550	59.8000	5.0550	932.0000	2
75%	111.0000	552.5225	95.3033	801.0000	3522.5000	16.6000	62.8250	5.5975	1465.0000	2
max	208.0000	14671.8000	766.0800	10071.9000	33700.0000	41.4780	71.1000	6.5636	17100.0000	2

choosing the countries whose mean value is more than 91.61

In [226…

```
df1_final_list = Kmean[Kmean['child_mort']>91]
df1_final_list.shape
```

Out[226]:

```
(21, 11)
```

In [236…

```
df1_final_list.describe()
```

Out[236]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	
count	21.0000	21.0000	21.0000	21.0000	21.0000	21.0000	21.0000	21.0000	21.0000	21
mean	121.7048	1010.7922	83.5849	850.4614	3639.1905	11.2144	56.5476	5.3865	1708.1905	2
std	27.1645	3164.9888	159.5347	2143.0015	7039.6580	10.1995	7.4018	0.9669	3607.6773	0
min	92.1000	20.6052	17.7508	90.5520	609.0000	0.8850	32.1000	3.3000	231.0000	2
25%	108.0000	101.2860	26.7960	170.8680	1190.0000	4.1500	55.6000	5.1100	446.0000	2
50%	114.0000	161.4240	40.6341	279.9360	1820.0000	6.8100	57.7000	5.3400	708.0000	2
75%	130.0000	460.9800	64.6600	428.3140	2690.0000	17.2000	60.1000	6.1600	1200.0000	2
max	208.0000	14671.8000	766.0800	10071.9000	33700.0000	41.4780	68.2000	6.5636	17100.0000	2

now Mean value of income is 3639 now choose the countries less than this mean valueof this.

In [228…

```
df1_final_list1 = df1_final_list[df1_final_list['income']<3639]
df1_final_list1.shape
```

Out[228]:

```
(17, 11)
```

In [229…

```
df1_final_list1.describe()
```

Out[229]:

	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp	label
count	17.0000	17.0000	17.0000	17.0000	17.0000	17.0000	17.0000	17.0000	17.0000	17.0000
mean	123.7471	213.2798	43.9756	333.3082	1611.3529	7.9897	55.3353	5.4151	698.3529	2.0000
std	29.1763	187.1063	26.6780	274.7639	781.5389	6.4973	7.6932	0.9839	345.5676	0.0000
min	93.6000	20.6052	17.7508	90.5520	609.0000	0.8850	32.1000	3.3000	231.0000	2.0000
25%	108.0000	81.5030	26.7960	170.2000	918.0000	2.9700	55.0000	5.1100	419.0000	2.0000
50%	114.0000	137.2740	38.7550	248.5080	1430.0000	5.4500	57.3000	5.3400	648.0000	2.0000
75%	137.0000	290.8200	52.2690	390.1950	1930.0000	12.3000	58.0000	6.2600	897.0000	2.0000
max	208.0000	617.3200	129.8700	1181.7000	3320.0000	20.8000	68.2000	6.5636	1310.0000	2.0000

Now Mean value of GDP is 698. Lets choose the countries less than this mean value

In [230…

```
df_final_list2 = df1_final_list1[df1_final_list1['gdpp']<698]
df_final_list2.shape
```

Out[230]: (10, 11)

In [231…

```
df_final_list2
```

Out[231]:

	country	child_mort	exports	health	imports	income	inflation	life_expec	total_fer	gdpp
66	Haiti	208.0000	101.2860	45.7442	428.3140	1500.0000	5.4500	32.1000	3.3300	662.0000
132	Sierra Leone	160.0000	67.0320	52.2690	137.6550	1220.0000	17.2000	55.0000	5.2000	399.0000
31	Central African Republic	149.0000	52.6280	17.7508	118.1900	888.0000	2.0100	47.5000	5.2100	446.0000
112	Niger	123.0000	77.2560	17.9568	170.8680	814.0000	2.5500	58.8000	6.5636	348.0000
37	Congo, Dem. Rep.	116.0000	137.2740	26.4194	165.6640	609.0000	20.8000	57.5000	6.5400	334.0000
25	Burkina Faso	116.0000	110.4000	38.7550	170.2000	1430.0000	6.8100	57.9000	5.8700	575.0000
64	Guinea-Bissau	114.0000	81.5030	46.4950	192.5440	1390.0000	2.9700	55.6000	5.0500	547.0000
63	Guinea	109.0000	196.3440	31.9464	279.9360	1190.0000	16.1000	58.0000	5.3400	648.0000
106	Mozambique	101.0000	131.9850	21.8299	193.5780	918.0000	7.6400	54.5000	5.5600	419.0000
26	Burundi	93.6000	20.6052	26.7960	90.5520	764.0000	12.3000	57.7000	6.2600	231.0000

Final List of countries which are in need of the aid based on socio-economic factors

In [232…

```
A_countries=df_final_list2['country']
A_countries=A_countries.reset_index(drop=True)
A_countries
```

Out[232]:
```
0    Haiti
1    Sierra Leone
2    Central African Republic
3    Niger
4    Congo, Dem. Rep.
5    Burkina Faso
6    Guinea-Bissau
7    Guinea
8    Mozambique
9    Burundi
Name: country, dtype: object
```

From the EDA performed we could see that Income, GDP and child Mortality are the major three variables need to be focused In K means clustering we got Cluster 2 is having very Low income, very Low GDP but High child mortality. So we concluded that countries under cluster 2 are in need of aid. In Hierarchical clustering we saw that Cluster 0 is having the High child mortality, low GDP and very Low child mortality. The clusters formed in Hierarchical clustering were not that good. So we went on to consider cluster formed in K means clustering. And got top five countries with High child mortality,Low GDP and Low income Then we looked for the countries based on socio economic factors

In [233…
```
print('Top 5 Countries based on K means clustering:')
Kmean[['country']].head()
```

```
Top 5 Countries based on K means clustering:
```

Out[233]:

	country
66	Haiti
132	Sierra Leone
32	Chad
31	Central African Republic
97	Mali

In [234…
```
print('Countries based on socio economic and health factors:')
A_countries
```

```
Countries based on socio economic and health factors:
```

Out[234]:
```
0    Haiti
1    Sierra Leone
2    Central African Republic
3    Niger
4    Congo, Dem. Rep.
5    Burkina Faso
6    Guinea-Bissau
7    Guinea
8    Mozambique
9    Burundi
Name: country, dtype: object
```

In [ ]: